UNIVERSIDAD DE LA RIOJA

# TRABAJO FIN DE ESTUDIOS

Título

## Aplicación para la gestión de peticiones de recursos de movilidad hospitalaria.


Autor/es

SERGIO GARRIDO LÓPEZ

Director/es

BEATRIZ PÉREZ VALLE

Facultad

Facultad de Ciencia y Tecnología

Titulación

Grado en Ingeniería Informática

Departamento

MATEMÁTICAS Y COMPUTACIÓN


Curso académico

2019-20




publicaciones.unirioja.es
E-mail: publicaciones@unirioja.es


**Facultad de Ciencia y Tecnología**

# TRABAJO FIN DE GRADO

Grado en Ingeniería Informática

Aplicación para la gestión de peticiones de recursos de movilidad hospitalaria

Realizado por:

Sergio Garrido López

Tutelado por:

Beatriz Pérez Valle

**Logroño, junio de 2020**

# Resumen

El presente proyecto fue propuesto por Javier Virto, jefe de proyectos de la oficina del Grupo GFI en La Rioja, y tiene como objetivo desarrollar una aplicación departamental que permita la recepción y gestión de peticiones de recursos de movilidad hospitalaria.

En particular, se ha desarrollado una aplicación web que permitirá gestionar las peticiones recibidas mediante listados, asignación y desasignación de recursos, así como el seguimiento centralizado de la flota de ambulancias.

Para realizar este trabajo se han aplicado tanto los conocimientos y técnicas adquiridos durante la realización de los estudios del Grado en Ingeniería Informática y la estancia en la empresa donde previamente realicé las prácticas, como otras tecnologías que he tenido que ir descubriendo durante el desarrollo de este proyecto. El lenguaje de programación principal utilizado ha sido C#.

# Abstract

The present project was proposed by Javier Virto, project manager of the GFI Group office in La Rioja, and aims to develop a departmental application that allows the reception and management of requests for hospital mobility resources.

In particular, a web application has been developed that will allow the management of requests received through listings, allocation and dis-allocation of resources, as well as the centralized monitoring of the ambulance fleet.

In order to carry out this work, both the knowledge and techniques acquired during the studies of the Degree in Computer Engineering and the stay in the company where I previously carried out the internship, as well as other technologies that I have had to discover during the development of this project, have been applied. The main programming language used has been C#.

# Tabla de contenido

# 1. Introducción

## 1.1. Antecedentes

GFI España [1] pertenece al Grupo GFI, una multinacional de consultoría, *outsourcing* e integración de sistemas en Tecnologías de la Información, que está presente en España desde 1998, con varias oficinas repartidas por todo el territorio nacional.

Una de esas oficinas está situada en Logroño, donde hay una *Software Factory* centrada en el sector de Sanidad que se dedica al desarrollo de evolutivos para una aplicación web que se utiliza en diversos hospitales en el ámbito nacional. Esto, unido al hecho de que realizara las prácticas de empresa en esta oficina, ha motivado que el proyecto se encuentre en el contexto del ámbito sanitario.

La idea del proyecto surge a partir de la necesidad de los hospitales de La Rioja de tener más localizada y centralizada la información relacionada con los traslados de pacientes en ambulancias ordinarias. Esta información recoge los datos básicos de los pacientes, las ambulancias y los traslados.

## 1.2. Objetivo del proyecto

Para realizar este proyecto vamos a suponer la existencia de un sistema externo (fuera del alcance del Trabajo de Fin de Grado) que se comunicará con la aplicación a desarrollar a través del motor de integración *Mirth* (motor multiplataforma que permite el envío bidireccional de mensajes *Health Level Seven* o *HL7* entre sistemas y aplicaciones) y utilizando mensajería de peticiones basada en estándar *HL7* (conjunto de estándares cuyo principal objetivo es especificar mensajería para la comunicación de información clínica).

El objetivo del presente proyecto es desarrollar una aplicación departamental que permita la recepción y gestión de peticiones de recursos de movilidad hospitalaria, entendiéndose como recursos cada una de las ambulancias del hospital. El motor *Mirth* se encargará de la recepción de estas peticiones, las cuales se encontrarán en el formato *HL7*. Este motor detectará las entradas de nuevas peticiones y las depositará en otro lugar distinto con un nuevo formato, cuya comprensión y manejo por parte del usuario no sea tan compleja como el formato *HL7*. La aplicación obtendrá las peticiones con el nuevo formato y las insertará en la base de datos de la aplicación. En cuanto a la gestión, nos centraremos en añadir, eliminar y/o modificar las peticiones ya existentes, así como aplicar distintos filtros de búsqueda de peticiones. También se quiere tener un seguimiento de la flota de ambulancias para poder conocer si están haciendo un traslado en un momento determinado, en cuyo caso informaremos del origen, el lugar de llegada y una hora de llegada estimada.

Del mismo modo que con las peticiones, también se podrán añadir nuevos recursos y eliminar o modificar los ya existentes. En ambos casos, tanto peticiones como recursos, se ofrecerá al usuario la posibilidad de obtener la información de una manera más ampliada.

## 1.3. Alcance del proyecto

El funcionamiento que se pretende dar a la aplicación es el siguiente: un usuario del hospital mandará, a través del sistema externo comentado previamente, una petición *HL7* que contendrá la información del traslado. Para la realización de este proyecto, y como se ha mencionado en el apartado *1.2. Objetivo del proyecto,* vamos a suponer la existencia de un usuario del hospital que manda la petición *HL7*, por lo que en vez de usar el motor *Mirth* de la manera más habitual, que consiste en detectar peticiones mediante la escucha de puertos y usando conexiones *TCP*, emplearemos la otra opción posible para la recepción de peticiones, que consiste en leer ficheros de una carpeta local. Para ello, iremos almacenando en esa carpeta local ficheros en formato *.orm* (formato de los mensajes de peticiones *HL7*). Tras la recepción, *Mirth* aplicará las transformaciones necesarias al mensaje para generar un nuevo archivo con otro formato que pueda ser mejor manejado por el usuario. Una vez realizada la transformación, *Mirth* depositará el mensaje con el nuevo formato en otra carpeta local, donde habrá un *listener* de la aplicación que estará atento para captar las nuevas peticiones e insertarlas en la base de datos.

La aplicación debe arrancar con una pantalla de *Login*, la cual dará acceso a la pantalla de *Home* de la aplicación. En esta pantalla se dará a elegir entre *Peticiones* o *Recursos,* ya sea a través de un menú de navegabilidad superior o a través de dos iconos, uno para cada opción. Independientemente de la opción elegida, se cargará una pantalla que muestre listas de peticiones o recursos según un orden predeterminado. En esa misma pantalla se debe dar la opción de filtrar la búsqueda, así como de añadir (de forma manual mediante formularios), modificar y eliminar peticiones o recursos, tras lo cual, se actualizarán de forma automática los resultados de la búsqueda. Además de estas opciones, también se dará la posibilidad de ver con más detalle el recurso o la petición seleccionada, ofreciendo incluso, en el caso de las peticiones, su descarga en formato PDF.

## 1.4. Metodología

Se ha decidido desarrollar el proyecto siguiendo una metodología ágil, más concretamente *Scrum* [2]. Los principales motivos de esta elección son que *Scrum* es la metodología que se emplea en la mayor parte de trabajos de larga duración desarrollados en la Universidad, y también porque es la metodología de trabajo utilizada en la empresa en la que se desarrolla este proyecto.

En *Scrum* se realizan entregas parciales y funcionales del producto final, priorizadas por el beneficio que aportan al receptor del proyecto.

Según esta metodología, un proyecto se desarrolla en bloques de tiempo cortos y fijos, llamados *sprints* o iteraciones. Un sprint representa un bloque de tiempo de entre 1 y 4 semanas, durante el cual se crea un incremento de producto "terminado" y utilizable.

En el marco de trabajo de *Scrum*, o *Scrum Framework*, podemos distinguir 3 grandes bloques: *Roles*, *Artefactos* y *Reuniones*.

El equipo *Scrum* está formado por los siguientes *Roles*:

***Product Owner*** o dueño del producto, que representa al cliente. Se encarga de dar prioridad a las historias de usuario y ubicarlas en la lista de requisitos del producto.

***Scrum Master***, que es la figura encargada de liderar el equipo.

***Equipo,*** que tiene la responsabilidad de entregar el producto.

En este proyecto en particular, los roles para las personas que intervienen son:

*Product Owner:* Javier Virto Pérez

*Scrum Master:* Sergio Garrido López

*Equipo:* Sergio Garrido López

En el bloque de *Artefactos* distinguimos entre:

***Product Backlog:*** lista con todos los trabajos deseados en el proyecto, priorizados por el *Product Owner*.

***Sprint Backlog:*** contiene las tareas que se van a desarrollar durante el sprint.

***Incremento:*** parte del producto producida en un sprint.

A lo largo de un Sprint se llevan a cabo 4 tipos de *Reuniones*:

***Sprint Planning***: tiene lugar al inicio de cada sprint y en ella se seleccionan los requisitos que se van a desarrollar en ese sprint y se planifica la iteración.

***Daily Meeting***: cada miembro del equipo expone al resto del equipo la evolución respecto del día anterior y la evolución prevista para el día siguiente.

***Sprint Review***: tiene lugar al final de cada sprint. El equipo presenta una versión del producto.

***Sprint Retrospective***: tiene lugar al final de cada sprint. Sirve para que los integrantes del equipo den sus impresiones sobre el sprint que acaba de terminar.

A pesar de que *Scrum* está pensado para trabajar en equipo, la realización de este proyecto es individual. Por este motivo, el equipo está compuesto solo por 1 persona y es por eso también por lo que no habrá reuniones de equipo, solo habrá *Sprint Planning* y *Sprint Review.*

## 1.5. Tecnologías a utilizar

Las tecnologías que se van a utilizar en este proyecto son:

***Health Level Seven (HL7)*** [3]: es un conjunto de estándares cuyo principal objetivo es especificar mensajería para el intercambio y el desarrollo de información clínica entre sistemas informáticos.

*HL7* define un mensaje abstracto compuesto por una colección de segmentos, dispuestos en una secuencia predeterminada.

***Mirth Connect*** [4]: es un motor multiplataforma que permite el envío bidireccional de mensajes entre sistemas y aplicaciones de forma fácil y rápida.

*Mirth Connect* permite la construcción de canales para el intercambio de mensajes entre sistemas y aplicaciones, además de realizar conversiones o transformaciones de formato dentro de un canal.

***Microsoft Visual Studio*** [5]: "El entorno de desarrollo integrado de Visual Studio es un panel de inicio creativo que se puede usar para editar, depurar y compilar código y, después, publicar una aplicación. Más allá del editor estándar y el depurador que proporcionan la mayoría de IDE, Visual Studio incluye compiladores, herramientas de finalización de código, diseñadores gráficos y muchas más características para facilitar el proceso de desarrollo de software."

La elección de Visual Studio se basa en que es el IDE con el que se trabaja en la empresa en la que he realizado tanto las prácticas como este TFG, además de ser empleado en algunas asignaturas en la Universidad, por lo que es uno de los IDE con el que más familiarizado estoy.

***SQL Server***: es un sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft.

La elección de SQL Server se basa en que es el sistema de gestión de bases de datos (SGBD) con el que se trabaja en GFI, además de ser SGBD que más se emplea en las asignaturas en la Universidad en las que se trabaja con Bases de Datos.

# 2. Planificación

## 2.1. Diagrama EDT del proyecto

La figura 2.1 representa la EDT del proyecto, la cual se detalla en el siguiente apartado.

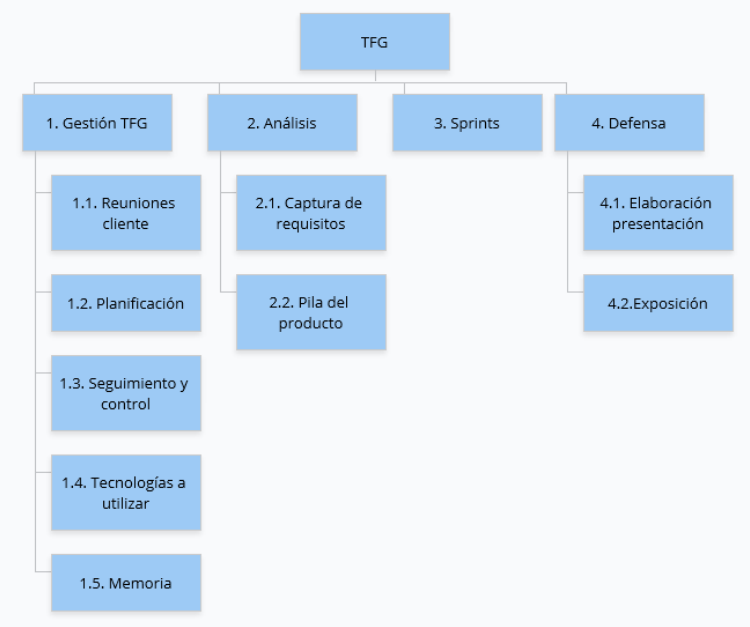



Figura 2.1. Diagrama EDT del proyecto.

## 2.2. Diccionario de la EDT

A continuación, se describen las actividades de los paquetes de trabajo del proyecto:

- **P1.1. Reuniones cliente:** Tarea que se alarga durante toda la realización del proyecto y que engloba desde las reuniones iniciales de análisis de requisitos hasta la reunión final para mostrar el proyecto finalizado, incluyendo también reuniones intermedias de los *Sprint Review*.
- **P1.2. Planificación:** Tarea en la que se hace la estimación de tiempo y dedicaciones del proyecto, marcando la división de éste en sprints y tareas, estableciendo igualmente la previsión de dedicación a cada uno de ellos.
- **P1.3. Seguimiento y control:** Seguimiento del desarrollo del proyecto, comparando las horas planificadas con las reales y teniendo en cuenta los posibles inconvenientes que puedan surgir, con las consiguientes replanificaciones.
- **P1.4. Tecnologías a utilizar:** Tarea en la que se incluye tanto las decisiones sobre el software que se va a utilizar en el proyecto, como el tiempo de investigación de funcionamiento de las tecnologías con las que no estoy familiarizado.
- **P1.5. Memoria:** Hace referencia a la redacción del documento donde se detalla el desarrollo del proyecto.

- **P2.1. Captura de requisitos:** Recopilación de los requisitos solicitados por el cliente y la posterior verificación del cliente.
- **P2.2. Pila del producto:** A partir de los requisitos recopilados, conformación de las historias de usuario que forman la pila del producto.
- **P3.1.1 Planificación del sprint**[1]**:** Aquí se incluyen las reuniones de *Sprint Planning* y la formación de la pila del Sprint, desglosada en el conjunto de tareas necesarias para implementar cada historia de usuario incluida en el sprint y tratando las posibles desviaciones.
- **P3.1.2 Implementación del sprint:** Desarrollo de la parte funcional del sprint.
- **P4.1. Elaboración presentación:** Elaboración de la presentación que se utilizará en la defensa del proyecto.
- **P4.2. Exposición:** Defensa del proyecto ante el tribunal.

## 2.3. Diagrama de Gantt

Para realizar la planificación, he tenido en cuenta que tengo que compaginar el desarrollo del TFG con la realización de una asignatura, y que también tengo un contrato de trabajo de 20 horas semanales en GFI.

Las tablas 2.2, 2.3, 2.4, 2.5, 2.6 y 2.7 representan el Gantt del proyecto. Los números entre paréntesis al lado de algunos paquetes de trabajo hacen referencia al número del sprint al cual corresponden.

| Cronograma – parte 1 | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Febrero | | | | | | | | | | | | | | | | | | |
| | | 10 | 11 | 12 | 13 | 14 | | 17 | 18 | 19 | 20 | 21 | | 24 | 25 | 26 | 27 | 28 | |
| P1.1 | Reuniones cliente | | | | | | | | | | | | | | | | | | | |
| P1.2 | Planificación | | | | | | | | | | | | | | | | | | | |
| P1.4 | Tecnologías a utilizar | | | | | | | | | | | | | | | | | | | |
| P1.5 | Memoria | | | | | | | | | | | | | | | | | | | |
| P2.1 | Captura de requisitos | | | | | | | | | | | | | | | | | | | |
| P2.2 | Pila del producto | | | | | | | | | | | | | | | | | | | |
| P3.1.1 | Planificación del sprint (1) | | | | | | | | | | | | | | | | | | | |
| P3.1.2 | Implementación del sprint (1) | | | | | | | | | | | | | | | | | | | |

Tabla 2.2. Diagrama Gantt del proyecto. Febrero.

| Cronograma – parte 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Marzo | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | 2 | 3 | 4 | 5 | 6 | | 9 | 10 | 11 | 12 | 13 | | 16 | 17 | 18 | 19 | 20 | | 23 | 24 | 25 | 26 | 27 | | 30 | 31 | | | |
| P1.1 | Reuniones cliente | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.3 | Seguimiento y control | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.5 | Memoria | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.1.2 | Implementación del sprint (1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.2.1 | Planificación del sprint (2) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.2.2 | Implementación del sprint (2) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Tabla 2.3. Diagrama Gantt del proyecto. Marzo.

[1] El trabajo descrito para este paquete de trabajo (P3.1.1) y el siguiente (P3.1.2) se aplica a los 5 sprints planificados para este proyecto.

| Cronograma – parte 3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Abril | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | 1 | 2 | 3 | | 6 | 7 | 8 | 9 | 10 | | 13 | 14 | 15 | 16 | 17 | | 20 | 21 | 22 | 23 | 24 | | 27 | 28 | 29 | 30 | | | |
| P1.1 | Reuniones cliente | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.3 | Seguimiento y control | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.5 | Memoria | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.2.2 | Implementación del sprint (2) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.3.1 | Planificación del sprint (3) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.3.2 | Implementación del sprint (3) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.4.1 | Planificación del sprint (4) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.4.2 | Implementación del sprint (4) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Tabla 2.4. Diagrama Gantt del proyecto. Abril.

| Cronograma – parte 4 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Mayo | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | 1 | | 4 | 5 | 6 | 7 | 8 | | 11 | 12 | 13 | 14 | 15 | | 18 | 19 | 20 | 21 | 22 | | 25 | 26 | 27 | 28 | 29 | | | | |
| P1.1 | Reuniones cliente | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.3 | Seguimiento y control | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.5 | Memoria | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.4.2 | Implementación del sprint (4) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.5.1 | Planificación del sprint (5) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.5.2 | Implementación del sprint (5) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Tabla 2.5. Diagrama Gantt del proyecto. Mayo.

| Cronograma – parte 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Junio | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | 1 | 2 | 3 | 4 | 5 | | 8 | 9 | 10 | 11 | 12 | | 15 | 16 | 17 | 18 | 19 | | 22 | 23 | 24 | 25 | 26 | | 29 | 30 | | | |
| P1.1 | Reuniones cliente | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.3 | Seguimiento y control | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P1.4 | Memoria | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P3.5.2 | Implementación del sprint (5) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| P4.1 | Elaboración presentación | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Tabla 2.6. Diagrama Gantt del proyecto. Junio.

| Cronograma – parte 6 | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Paquete de trabajo | | Julio | | | | | | | | | | | | | | |
| | | 1 | 2 | 3 | | 6 | 7 | 8 | 9 | 10 | | 13 | 14 | 15 | 16 | |
| P4.1 | Presentación | | | | | | | | | | | | | | | |
| P4.2 | Exposición | | | | | | | | | | | | | | | |

Tabla 2.7. Diagrama Gantt del proyecto. Julio.

## 2.4. Diagrama de Hitos

En el siguiente diagrama de hitos se muestra la relación de cada paquete de trabajo con la fecha prevista de finalización del mismo. La tabla 2.8 muestra los hitos del proyecto.

| Paquete de trabajo | | Fecha de finalización |
|---|---|---|
| P1.1 | Reuniones cliente | 5 junio |
| P1.2 | Planificación | 21 febrero |
| P1.3 | Seguimiento y control | 8 junio |
| P1.4 | Tecnologías a utilizar | 13 febrero |
| P1.5 | Memoria | 10 junio |
| P2.1 | Captura de requisitos | 18 febrero |
| P2.2 | Pila del producto | 21 febrero |
| P3.1.1 | Planificación del sprint (1) | 24 febrero |
| P3.1.2 | Implementación del sprint (1) | 13 marzo |
| P3.2.1 | Planificación del sprint (2) | 16 marzo |
| P3.2.2 | Implementación del sprint (2) | 3 abril |
| P3.3.1 | Planificación del sprint (3) | 6 abril |
| P3.3.2 | Implementación del sprint (3) | 24 abril |
| P3.4.1 | Planificación del sprint (4) | 27 abril |
| P3.4.2 | Implementación del sprint (4) | 15 mayo |
| P3.5.1 | Planificación del sprint (5) | 18 mayo |
| P3.5.2 | Implementación del sprint (5) | 5 junio |
| P4.1 | Presentación | 3 julio |
| P4.2 | Exposición | 13-16 julio |

Tabla 2.8. Diagrama de Hitos del proyecto.

## 2.5. Tiempos estimados

La tabla 2.9 incluye la dedicación estimada para cada paquete de trabajo del proyecto.

| Paquete de trabajo | | Horas |
|---|---|---|
| P1.1 | Reuniones cliente | 15 |
| P1.2 | Planificación | 8 |
| P1.3 | Seguimiento y control | 8 |
| P1.4 | Tecnologías a utilizar | 15 |
| P1.5 | Memoria | 40 |
| P2.1 | Captura de requisitos | 5 |
| P2.2 | Pila del producto | 10 |
| P3.1.1 | Planificación del sprint (1) | 3 |
| P3.1.2 | Implementación del sprint (1) | 34 |
| P3.2.1 | Planificación del sprint (2) | 3 |
| P3.2.2 | Implementación del sprint (2) | 36 |
| P3.3.1 | Planificación del sprint (3) | 3 |
| P3.3.2 | Implementación del sprint (3) | 30 |
| P3.4.1 | Planificación del sprint (4) | 3 |
| P3.4.2 | Implementación del sprint (4) | 32 |
| P3.5.1 | Planificación del sprint (5) | 3 |
| P3.5.2 | Implementación del sprint (5) | 34 |
| P4.1 | Presentación | 15 |
| P4.2 | Exposición | 1 |
| **Proyecto** | **TFG** | **300** |

Tabla 2.9. Dedicaciones estimadas por paquete de trabajo (redondeadas en horas).

# 3. Desarrollo

En este capítulo de desarrollo se explicará el trabajo realizado en cada uno de los sprints en los que está dividido el proyecto, pero primero se describirá la planificación de dichos sprints.

## 3.1. Planificación de los sprints

Como se ha adelantado anteriormente, se ha decidido que el desarrollo de este proyecto se va a realizar en 5 sprints. Cada uno de estos sprints tendrá 3 semanas de duración. A la hora de repartir la carga de historias de usuario entre los distintos sprints se tendrá en cuenta el valor que proporcione el producto al cliente al final de cada sprint.

### 3.1.1. Product Backlog

Antes de empezar a definir la pila del producto, primero vamos a definir qué es un punto de historia. Un punto de historia es el esfuerzo estimado para la realización de una tarea. Para medir el esfuerzo estimado en este proyecto, se ha utilizado la secuencia de Fibonacci (1, 2, 3, 5, 8, 13, 21, infinito), que es la metodología más recurrente a la hora de estimar el esfuerzo de las historias de usuario. En la secuencia de Fibonacci se asigna el valor 1 a las tareas pequeñas y el valor infinito a las tareas grandes que deben subdividirse en historias más pequeñas.

Para clasificar las historias de usuario se va a emplear el método MoSCoW, en el que las historias de usuario se clasifican en 4 niveles:

***Must:*** funcionalidad que el proyecto debe tener.

***Should***: funcionalidad que el proyecto debería tener.

***Could***: sería conveniente tener esta funcionalidad. Es en realidad un deseo.

***Won´t***: no está en los planes tener esta funcionalidad en este momento.

En este proyecto no habrá historias de usuario que estén en la clasificación de *Won´t*, ya que el cliente ha definido claramente los requisitos de la aplicación, así como su importancia. En cuanto a la estimación de puntos de historia, y tras haberlo consensuado con el cliente, en este proyecto la estimación está comprendida entre 1 y 8 puntos de historia.

## Historias de usuario imprescindibles (must have)

| **Historia de usuario** | |
|---|---|
| **Login** | 2 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder entrar a la aplicación con mi nombre de usuario y contraseña. | |
| Aceptación | |
| El usuario accederá a la aplicación si los datos son válidos y se mostrará un mensaje de error en caso de no serlo. | |

| **Historia de usuario** | |
|---|---|
| **Home** | 1 puntos de historia |
| Tarjeta | |
| Como usuario, quiero disponer de una pantalla de *Home* donde pueda tener navegabilidad a las principales partes de la aplicación. | |
| Aceptación | |
| El usuario accederá a la pantalla de *Home* tras un *Login* correcto o a través de un menú de navegabilidad superior. | |

| **Historia de usuario** | |
|---|---|
| **Ver todas las peticiones** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver todas las peticiones. | |
| Aceptación | |
| Se mostrará al usuario una pantalla con una lista de peticiones, que serán mostradas en varias páginas si es necesario. | |

| **Historia de usuario** | |
|---|---|
| **Filtrar peticiones** | 8 puntos de historia |
| Tarjeta | |
| Como usuario, quiero disponer de un filtro para acotar más la búsqueda de peticiones. | |
| Aceptación | |
| Se mostrará al usuario una pantalla con una lista de peticiones que cumplan los criterios solicitados por el usuario. Esta lista será mostrada en varias páginas si es necesario. | |

| **Historia de usuario** | |
|---|---|
| **Añadir petición** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder añadir una nueva petición. | |
| Aceptación | |
| Una vez validada la nueva petición, se actualizará la búsqueda de peticiones, donde la nueva petición podrá ser visualizada si cumple con los criterios de búsqueda. Si la nueva petición no es válida, se mostrará un mensaje de error. | |

| **Historia de usuario** | |
|---|---|
| **Modificar petición** | 1 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder modificar los datos de una petición. | |
| Aceptación | |
| Una vez validados los datos, se actualizará la búsqueda de peticiones, donde la petición modificada podrá ser visualizada si cumple con los criterios de búsqueda. Si los datos no son válidos, se mostrará un mensaje de error. | |

| Historia de usuario | |
|---|---|
| **Ver todos los recursos** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver todos los recursos. | |
| Aceptación | |
| Se mostrará al usuario una pantalla con una lista de los recursos, que serán mostrados en varias páginas si es necesario. | |

| Historia de usuario | |
|---|---|
| **Filtrar recursos** | 8 puntos de historia |
| Tarjeta | |
| Como usuario, quiero disponer de un filtro para acotar más la búsqueda de recursos. | |
| Aceptación | |
| Se mostrará al usuario una pantalla con una lista de recursos que cumplan los criterios solicitados por el usuario. Esta lista será mostrada en varias páginas si es necesario. | |

| Historia de usuario | |
|---|---|
| **Añadir recurso** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder añadir un nuevo recurso. | |
| Aceptación | |
| Una vez validado el nuevo recurso, se actualizará la búsqueda de recursos, donde el nuevo recurso podrá ser visualizado si cumple con los criterios de búsqueda. Si el nuevo recurso no es válido, se mostrará un mensaje de error. | |

| Historia de usuario | |
|---|---|
| **Modificar recurso** | 1 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder modificar los datos de un recurso. | |
| Aceptación | |
| Una vez validados los datos, se actualizará la búsqueda de recursos, donde el recurso modificado podrá ser visualizado si cumple con los criterios de búsqueda. Si los datos no son válidos, se mostrará un mensaje de error. | |

| Historia de usuario | |
|---|---|
| **Eliminar petición** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder eliminar una petición. | |
| Aceptación | |
| Una vez eliminada la petición, se actualizará la búsqueda de peticiones, donde la petición eliminada no podrá ser visualizada. | |

| Historia de usuario | |
|---|---|
| **Eliminar recurso** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder eliminar un recurso. | |
| Aceptación | |
| Una vez eliminado el recurso, se actualizará la búsqueda de recursos, donde el recurso eliminado no podrá ser visualizado. | |

## Historias de usuario importantes (should have)

| **Historia de usuario** | |
|---|---|
| **Ampliar detalles petición** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver con detalle una petición de la lista de peticiones. | |
| Aceptación | |
| Al usuario se le mostrará una nueva pantalla con los datos de la petición seleccionada. | |

| **Historia de usuario** | |
|---|---|
| **Ampliar detalles recurso** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver con detalle un recurso de la lista de recursos. | |
| Aceptación | |
| Al usuario se le mostrará una nueva pantalla con los datos del recurso seleccionado. | |

| **Historia de usuario** | |
|---|---|
| **Identificar paciente petición** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver la información del paciente asociado a la petición. | |
| Aceptación | |
| Al usuario se le mostrará una nueva pantalla con los datos del paciente asociado a la petición seleccionada. | |

| **Historia de usuario** | |
|---|---|
| **Identificar recurso petición** | 3 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder ver la información del recurso asociado a la petición. | |
| Aceptación | |
| Al usuario se le mostrará una nueva pantalla con los datos del recurso asociado a la petición seleccionada. | |

## Historias de usuario buenas (could have)

| **Historia de usuario** | |
|---|---|
| **Descargar información petición en PDF** | 8 puntos de historia |
| Tarjeta | |
| Como usuario, quiero poder descargar la información de una petición en formato PDF. | |
| Aceptación | |
| Se descargará un archivo PDF con los datos de una petición. | |

Entre todas las historias de usuario suman 59 puntos de historia, por lo que, teniendo en cuenta que habrá 5 sprints, se intentarán distribuir en 12 puntos de historia por sprint.

Además del incremento del producto, cada sprint también estará compuesto por reuniones con el cliente, documentación de la memoria y el tratamiento de posibles desviaciones.

### 3.1.2. Sprint 1

#### 3.1.2.1. Sprint Backlog

El primer sprint estará compuesto por las siguientes historias de usuario:

- Login (2 puntos de historia)
- Home (1 punto de historia)
- Ver todos los recursos (3 puntos de historia)
- Ver todas las peticiones (3 puntos de historia)

La estimación es de 9 puntos de historia. En este sprint no se cumple la dedicación ideal de 12 puntos de historia por sprint porque, al ser el primer sprint, hay que definir la base de datos y crear la infraestructura de la aplicación y prepararla para que esté atenta a captar los mensajes del motor *Mirth*.

#### 3.1.2.2. Descomposición de las historias de usuario en tareas

En la tabla 3.1 se puede ver la lista de tareas que es necesario realizar para cumplir los objetivos planificados para el sprint.

| Historia de usuario | Tarea | Descripción |
|---|---|---|
| | Tarea 1 | Diseño de la base de datos |
| | Tarea 2 | Población de la base de datos |
| | Tarea 3 | Diseño de la infraestructura de la aplicación |
| | Tarea 4 | Creación de la infraestructura de la aplicación |
| Login | Tarea 5 | Diseño de la pantalla de *Login* |
| | Tarea 6 | Desarrollo de la pantalla de *Login* |
| | Tarea 7 | Pruebas |
| Home | Tarea 8 | Diseño de la pantalla de *Home* |
| | Tarea 9 | Desarrollo de la pantalla de *Home* |
| | Tarea 10 | Pruebas |
| Ver todos los recursos | Tarea 11 | Diseño de la pantalla de *Recursos* |
| | Tarea 12 | Desarrollo de la pantalla de *Recursos* |
| | Tarea 13 | Pruebas |
| Ver todas las peticiones | Tarea 14 | Diseño de la pantalla de *Peticiones* |
| | Tarea 15 | Desarrollo de la pantalla de *Peticiones* |
| | Tarea 16 | Pruebas |
| | Tarea 17 | Integración de *Mirth* |

Tabla 3.1. Lista de tareas del primer sprint

### 3.1.3. Sprint 2

#### 3.1.3.1. Sprint Backlog

El segundo sprint estará compuesto por las siguientes historias de usuario, siendo la estimación de 14 puntos de historia:

- Añadir petición (3 puntos de historia)
- Añadir recurso (3 puntos de historia)
- Filtrar peticiones (8 puntos de historia)

En este sprint tampoco se cumple la dedicación ideal de 12 puntos de historia por sprint porque hay que recuperar el trabajo, en cuanto a historias de usuario, que no se pudo realizar en el sprint anterior debido a las tareas de configuración.

#### 3.1.3.2. Descomposición de las historias de usuario en tareas

En la tabla 3.2 se puede ver la lista de tareas que es necesario realizar para cumplir los objetivos planificados para el sprint.

| Historia de usuario | Tarea | Descripción |
|---|---|---|
| Añadir petición | Tarea 1 | Diseño de la pantalla de *Añadir Petición* |
| | Tarea 2 | Desarrollo de la pantalla de *Añadir Petición* |
| | Tarea 3 | Pruebas |
| Añadir recurso | Tarea 4 | Diseño de la pantalla de *Añadir Recurso* |
| | Tarea 5 | Desarrollo de la pantalla de *Añadir Recurso* |
| | Tarea 6 | Pruebas |
| Filtrar peticiones | Tarea 7 | Añadir filtro a la pantalla inicial de *Peticiones* |
| | Tarea 8 | Pruebas |

Tabla 3.2. Lista de tareas del segundo sprint.

### 3.1.4. Sprint 3

#### 3.1.4.1. Sprint Backlog

El tercer sprint estará compuesto por las siguientes historias de usuario, siendo la estimación de 12 puntos de historia:

- Filtrar recursos (8 puntos de historia)
- Modificar petición (1 punto de historia)
- Ampliar detalles petición (3 puntos de historia)

#### 3.1.4.2. Descomposición de las historias de usuario en tareas

En la Tabla 3.3. se puede ver la lista de tareas que es necesario realizar para cumplir los objetivos planificados para el sprint.

| Historia de usuario | Tarea | Descripción |
|---|---|---|
| Filtrar recursos | Tarea 1 | Añadir filtro a la pantalla de inicial de *Recursos* |
| | Tarea 2 | Pruebas |
| Modificar petición | Tarea 3 | Diseño de la pantalla de *Modificar Petición* |
| | Tarea 4 | Desarrollo de la pantalla de *Modificar Petición* |
| | Tarea 5 | Pruebas |
| Ampliar detalles petición | Tarea 6 | Diseño de la pantalla de *Ampliar detalles Petición* |
| | Tarea 7 | Desarrollo de la pantalla de *Ampliar detalles Petición* |
| | Tarea 8 | Pruebas |

Tabla 3.3. Lista de tareas del tercer sprint.

### 3.1.5. Sprint 4

#### 3.1.5.1. Sprint Backlog

La estimación del cuarto sprint es de 12 puntos de historia y está compuesto por:

- Eliminar petición (3 puntos de historia)
- Eliminar recurso (3 puntos de historia)
- Identificar recurso petición (3 puntos de historia)
- Identificar paciente petición (3 puntos de historia)

#### 3.1.5.2. Descomposición de las historias de usuario en tareas

En la tabla 3.4 se puede ver la lista de tareas que es necesario realizar para cumplir los objetivos planificados para el sprint.

| Historia de usuario | Tarea | Descripción |
|---|---|---|
| Eliminar Petición | Tarea 1 | Diseño de la pantalla de *Eliminar Petición* |
| | Tarea 2 | Desarrollo de la pantalla de *Eliminar Petición* |
| | Tarea 3 | Pruebas |
| Eliminar Recurso | Tarea 4 | Diseño de la pantalla de *Eliminar Recurso* |
| | Tarea 5 | Desarrollo de la pantalla de *Eliminar Recurso* |
| | Tarea 6 | Pruebas |
| Identificar Recurso Petición | Tarea 7 | Diseño de la pantalla de *Identificar Recurso Petición* |
| | Tarea 8 | Desarrollo de la pantalla de *Identificar Recurso Petición* |
| | Tarea 9 | Pruebas |
| Identificar Paciente Petición | Tarea 10 | Diseño de la pantalla de *Identificar Paciente Petición* |
| | Tarea 11 | Desarrollo de la pantalla de *Identificar Paciente Petición* |
| | Tarea 12 | Pruebas |

Tabla 3.4 Lista de tareas del cuarto sprint.

### 3.1.6. Sprint 5

#### 3.1.6.1. Sprint Backlog

La estimación del quinto sprint es de 12 puntos de historia y está compuesto por:

- Modificar recurso (1 punto de historia)
- Ampliar detalles recurso (3 puntos de historia)
- Descargar información petición PDF (8 puntos de historia)

#### 3.1.6.2. Descomposición de las historias de usuario en tareas

En la tabla 3.5 se puede ver la lista de tareas que es necesario realizar para cumplir los objetivos planificados para el sprint.

| Historia de usuario | Tarea | Descripción |
|---|---|---|
| Modificar recurso | Tarea 1 | Diseño de la pantalla de *Modificar Recurso* |
| | Tarea 2 | Desarrollo de la pantalla de *Modificar Recurso* |
| | Tarea 3 | Pruebas |
| Ampliar detalles recurso | Tarea 4 | Diseño de la pantalla de *Ampliar detalles Recurso* |
| | Tarea 5 | Desarrollo de la pantalla de *Ampliar detalles Recurso* |
| | Tarea 6 | Pruebas |
| Descargar información petición en PDF | Tarea 7 | Añadir la opción de descarga a la pantalla de *Ampliar detalles Petición* |
| | Tarea 8 | Desarrollo de la funcionalidad |
| | Tarea 9 | Pruebas |

Tabla 3.5. Lista de tareas del quinto sprint.

## 3.2. Sprint 1

### 3.2.1. Incremento

El incremento que se obtendrá tras la finalización del sprint será:

1. Diseño y población de la base de datos.
2. Pantalla de *Login.*
3. Pantalla de *Home.*
4. Pantalla con el listado de recursos.
5. Pantalla con el listado de peticiones.
6. Integración del motor *Mirth*.

## 3.2.2. Diseño de la base de datos

En la Figura 3.6, se puede ver un diagrama UML que representa el diseño de la base de datos.

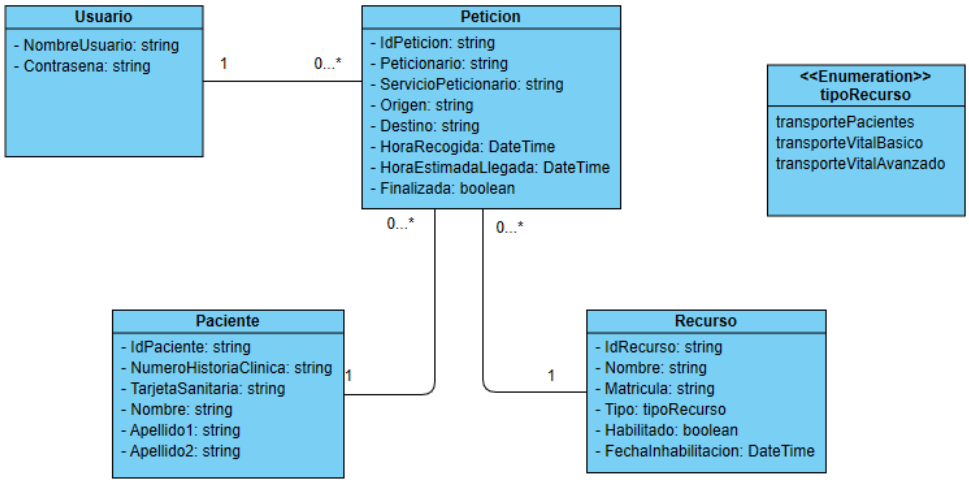



Figura 3.6. Diagrama UML.

Las tablas obtenidas a partir del diagrama anterior son las siguientes:

**PACIENTE**

| IdPaciente | NumeroHistoriaClinica | Tarjeta Sanitaria | Nombre | Apellido1 | Apellido2 |
|---|---|---|---|---|---|

Atributo (Tipo): IdPaciente (string), NumeroHistoriaClinica (String), TarjetaSanitaria (String), Nombre (String), Apellido1 (String), Apellido2 (String).

**RECURSO**

| IdRecurso | Nombre | Matricula | Tipo | Habilitado | Fecha Inhabilitación |
|---|---|---|---|---|---|

Atributo (Tipo): IdRecurso (Guid), Nombre (String), Matricula (String), Tipo (Enumerador), Habilitado (boolean), FechaInhabilitacion (DateTime)

**PETICION**

| IdPeticion | Peticionario | Servicio Peticionario | Origen | Destino | HoraRecogida | HoraEstimadaLLegada |
|---|---|---|---|---|---|---|

| IdPaciente | IdRecurso | Usuario | Finalizada |
|---|---|---|---|
| CE:Paciente | CE:Recurso | CE:Usuario | |

Atributo (Tipo): IdPeticion (Guid), Peticionario (String), ServicioPeticionario (String), Origen (String), Destino (String), HoraRecogida (DateTime), HoraEstimadaLlegada (DateTime), IdPaciente (Guid), IdRecurso (Guid), Usuario (Guid); Finalizada (boolean)

**USUARIO**

| NombreUsuario | Contrasena |
|---|---|

Atributo (Tipo): NombreUsuario (String), Contrasena (String).

## 3.2.3. Arquitectura de la aplicación

La aplicación se desarrollará siguiendo el Modelo Vista Controlador. En la imagen 3.7 se puede ver en qué consiste este modelo.

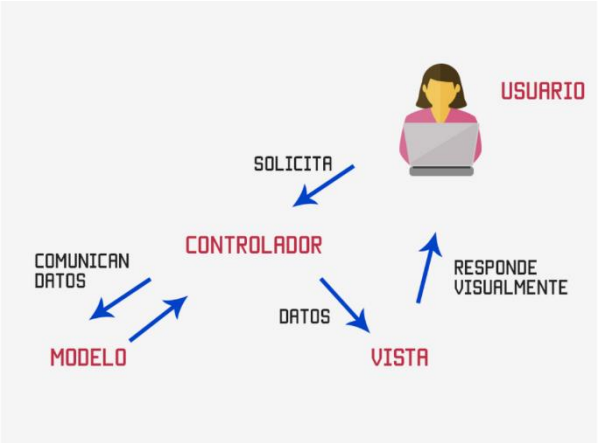



Imagen 3.7. Modelo Vista Controlador. Imagen obtenida de [6].

El **Modelo** es la capa donde se trabaja con los datos, por lo que contendrá mecanismos para acceder y actualizar la información. Los datos estarán habitualmente en una base de datos.

El **Controlador** actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos.

Las **Vistas** se encargan del diseño y de la presentación. Define cómo se deben mostrar los datos de la aplicación.

En la imagen 3.8 podemos ver cómo ha quedado definida la arquitectura de la aplicación.

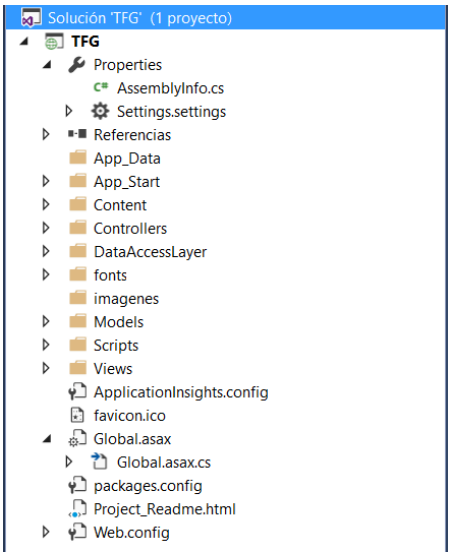



Imagen 3.8. Arquitectura de la aplicación. Imagen obtenida del proyecto de VisualStudio

En la imagen se pueden identificar las capas propias del MVC (*Models, Views, Controllers*). Además de esto, el proyecto está compuesto por otras carpetas como son:

- *Content*, donde se incluyen los archivos de estilos destinados a la parte visual de las vistas, es decir, archivos *css.*
- *DataAccessLayer,* donde se encuentran los métodos necesarios para obtener los datos que hay en la base de datos.
- *Imágenes,* donde están contenidas las imágenes que se utilizan en la aplicación.
- *Scripts,* donde están las referencias a las librerías que se utilizan en las vistas para mostrar los datos.

## 3.2.4. Trabajo de desarrollo

### 3.2.4.1. Desarrollo de la pantalla de *Login*

La pantalla de *Login*, como se puede ver en la imagen 3.9, consiste en 2 campos que tienen que ser completados por el usuario. En caso de que los datos introducidos sean erróneos, se mostrará al usuario un mensaje indicando que el usuario o la contraseña no son correctos.

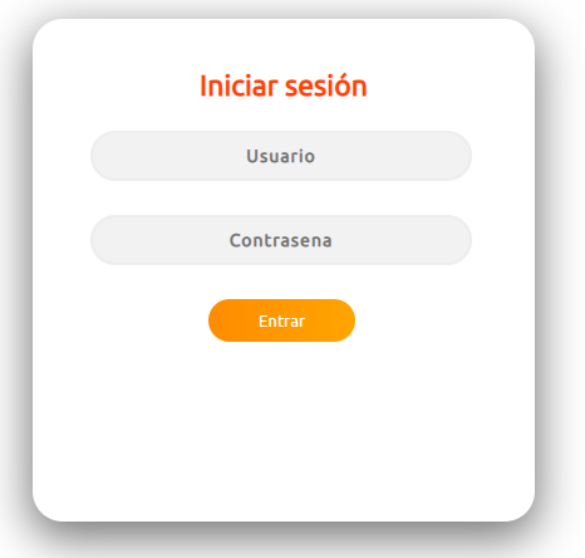



Imagen 3.9. Pantalla de *Login.*

Una vez introducidos los datos, se envían al controlador, donde se comprueban. En el caso de que los datos sean correctos, se envía al usuario a la pantalla de *Home*. En el caso contrario, se devuelve al usuario a la pantalla de *Login* mostrando un mensaje de error.

Para hacer las pruebas, se han considerado cuatro casuísticas:

1. Introducir nombre de usuario incorrecto y contraseña correcta.
2. Introducir nombre de usuario correcto y contraseña incorrecta.
3. Introducir nombre de usuario incorrecto y contraseña incorrecta.
4. Introducir nombre de usuario correcto y contraseña correcta.

Para los tres primeros casos se ha obtenido el mismo resultado, que se puede observar en la imagen 3.10. Tras el último caso, el Login es aceptado y el usuario es dirigido a la pantalla de *Home.*

En caso de no ser completado alguno o ambos campos el Login también es incorrecto y se muestra el mismo mensaje de error.


Imagen 3.10. Login incorrecto.

### 3.2.4.2. Desarrollo de la pantalla de *Home*

La pantalla de *Home* está compuesta por un menú de navegación en la parte superior, un par de párrafos para describir mínimamente la aplicación y otro par de imágenes que tienen la misma función que los componentes del menú de navegación. El diseño final se puede ver en la imagen 3.11.

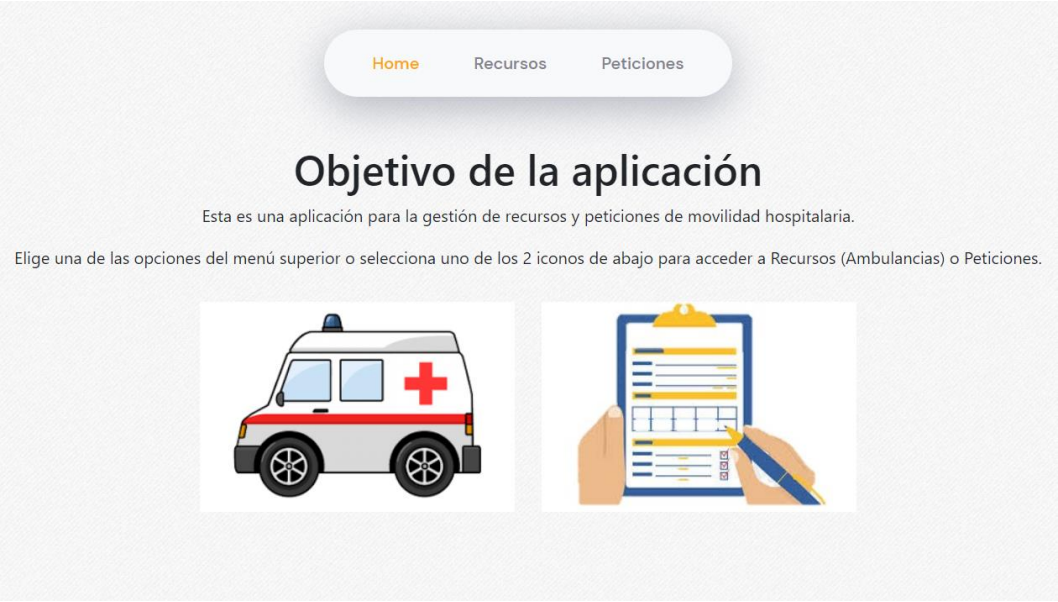



Imagen 3.11. Pantalla de *Home.*

Merece la pena destacar que el menú de navegación de la parte superior se mantendrá en la pantalla de consulta de Recursos y la pantalla de consulta de Peticiones.

En este caso no se han realizado pruebas, ya que hasta ahora el único momento en el que se llama a este controlador es tras el login.

### 3.2.4.3. Desarrollo de la pantalla de consulta de Recursos

Como ya se ha mencionado en el apartado *1.2. Objetivo del proyecto,* la aplicación dispondrá de una pantalla de gestión de recursos. Esta pantalla seguirá manteniendo el menú de navegación de la parte superior y será donde se muestren los recursos para su consulta y se ofrezcan las posibilidades de filtrar, añadir, modificar, eliminar y ver con mayor detalle los recursos. En la tarea asociada a este sprint nos centraremos en la consulta de recursos. Para mostrar todos los recursos registrados se ha utilizado una tabla que presentará el nombre del recurso, el tipo de recurso que es, y si está o no habilitado, pudiendo ser ordenados por nombre o tipo. La tabla mostrará 10 registros por página, por lo que en la parte inferior se incluirá la opción de navegación entre páginas.

Se ha desarrollado un controlador llamado *Recurso,* que recibe como parámetro el tipo de ordenación y puede o no recibir el número de página. En la imagen 3.12 se presenta el código del controlador.

```
[RoutePrefix("Recursos")]
public class RecursoController : Controller
{
    private RecursoContext db = new RecursoContext();

    [Route]
    public ViewResult MostrarRecursos(string tipoOrdenacion, int? pagina)
    {
        ViewBag.OrdenActual = tipoOrdenacion;
        ViewBag.OrdenarPorNombre = String.IsNullOrEmpty(tipoOrdenacion) ? "nombre_desc" : "";
        ViewBag.OrdenarPorTipo = tipoOrdenacion == "Tipo" ? "tipo_desc" : "Tipo";

        var recursos = from r in db.Recursos
                       select r;

        switch (tipoOrdenacion)
        {
            case "nombre_desc":
                recursos = recursos.OrderByDescending(r => r.Nombre);
                break;
            case "Tipo":
                recursos = recursos.OrderBy(r => r.Tipo);
                break;
            case "tipo_desc":
                recursos = recursos.OrderByDescending(r => r.Tipo);
                break;
            default:  // nombre ascendente
                recursos = recursos.OrderBy(r => r.Nombre);
                break;
        }

        int tamanoPagina = 10;
        int numeroPagina = (pagina ?? 1);
        return View(recursos.ToPagedList(numeroPagina, tamanoPagina));
    }
```

Imagen 3.12. Controlador *Recurso.*

Lo primero que hace es determinar qué criterio de ordenación tiene que seguir para mandar esta información a la vista. Después se obtienen todos los registros de la tabla *Recurso* de la base de datos y se ordenan siguiendo la condición que hemos recibido como parámetro para, posteriormente, definir el número de filas que tendrá la tabla y devolver los recursos paginados a la vista.

En la imagen 3.13. tenemos cuál es el aspecto visual de la aplicación para la pantalla de consulta de Recursos.

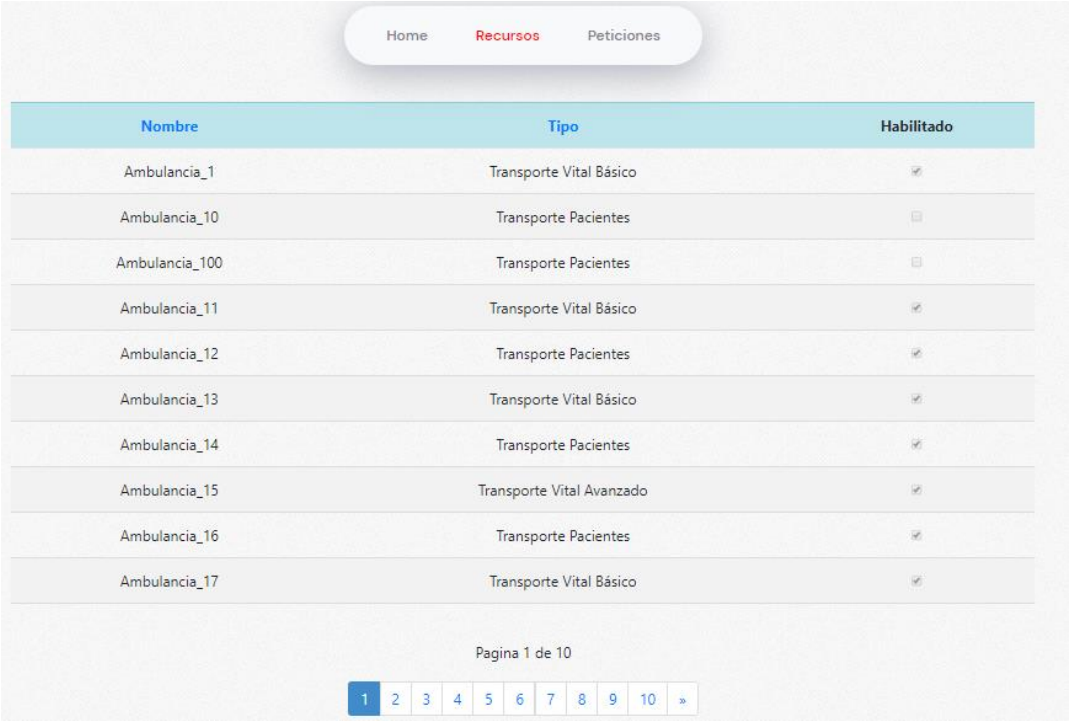



Imagen 3.13. Pantalla de consulta de Recursos.

Las pruebas para esta parte de la aplicación se han basado en comprobar que la vista se muestra de forma correcta tras ser llamada tanto desde el menú de navegación superior como desde el icono de recursos de la pantalla de *Home.* Una vez se ha cargado la vista, las pruebas han consistido en comprobar el correcto funcionamiento de los tipos de ordenación y del menú inferior de navegación entre páginas.

### 3.2.4.4. Desarrollo de la pantalla de consulta de Peticiones

La pantalla de gestión de peticiones es muy parecida a la de gestión de recursos, previamente explicada. Al igual que con los recursos, en este sprint nos centraremos en la consulta de peticiones. Una petición consiste en realizar una solicitud para que un determinado recurso traslade a un determinado paciente desde un origen hacia un destino.

Para mostrar todas las peticiones registradas, se ha utilizado una tabla que mostrará el origen y destino del recurso, el nombre del paciente y del recurso asignado a dicha petición, la hora de recogida y si la petición ha finalizado. En caso de que la petición no haya finalizado, se mostrará una hora estimada de llegada del recurso al destino. La tabla mostrará 10 registros por página, por lo que en la parte inferior se incluye la opción de navegación entre páginas.

La diferencia entre el controlador *Petición* y el controlador *Recurso* es el filtro de ordenación que recibe como parámetro. Recordemos que las peticiones tienen un recurso asociado, por lo que para cada petición nos interesa saber cuál es el lugar de partida del recurso y cuál es el destino final. Es por eso por lo que las peticiones se podrán ordenar según el lugar de origen y según el lugar de destino, tanto de forma ascendente como descendente. También se podrá ordenar por hora de recogida.

Después se obtienen todos los registros de la tabla *Petición* de la base de datos, para mostrarlos siguiendo el criterio de ordenación recibido como parámetro.

En la imagen 3.14. tenemos cuál es el aspecto visual de la aplicación para la pantalla de consulta de Peticiones.

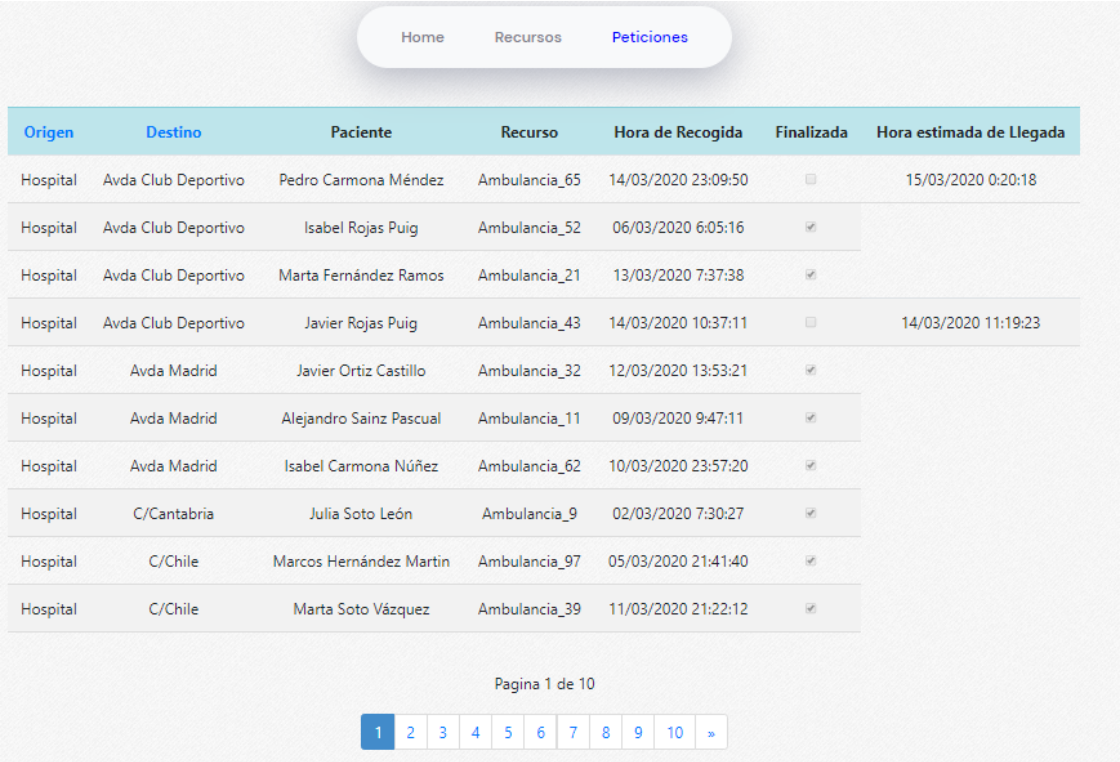



Imagen 3.14. Pantalla de consulta de Peticiones[2].

Las pruebas realizadas han sido equivalentes a las del caso anterior.

[2] Los datos pertenecientes tanto a esta imagen como a las siguientes donde se puedan apreciar nombres concretos son datos inventados.

### 3.2.4.5. Integración de Mirth

Trabajar directamente con mensajes *HL7* no es fácil. Un mensaje *HL7* es una cadena de texto cuya nomenclatura más habitual de expresión son las *pipes* (`|'). En la imagen 3.15. podemos ver un extracto de un mensaje *HL*7.

```
MSH|^~\&|sistemaExterno|idOrigenPeticion^lugarOrigenPeticion|lugarDestinoPeticion|seccionDestinoPeticion
PID|1|idPaciente|nhc^^^^~numeroTarjetaSanitaria^^TARJETA SANITARIA^HC~||apellidolPaciente^nombrePaciente
PV1|1||^habitacion^cama^centro||||nombreUsuario^apellidosSolicitante ^nombreSolicitante^^^^^ROL^^idRol~^
ORC|NW|idUsuario|||||^^^horaRecogida^horaEstimadaLlegada^|idUsuario|fechaSolicitudPeticion|nombreUsuario
OBR|1|idUsuario-numeroPeticion||idRecurso^nombreRecurso|||horaRecogida|horaEstimadaLlegada|tipoRecurso||.
NTE|0|1
```

Imagen 3.15. Extracto de un mensaje *HL7*.

Como se puede observar, resulta tedioso trabajar con este tipo de mensajes sin una herramienta que nos facilite su manipulación. En nuestro caso, emplearemos *Mirth Connect* para transformar los mensajes *HL7* recibidos en otro formato con el que nos sea más fácil de trabajar, en este caso *XML*. A continuación, vamos a explicar los principales componentes de la herramienta *Mirth Connect* [7]:

**Canal**

Un canal está compuesto por una fuente y, al menos, un destino. Tanto la fuente como los destinos son conectores. Los conectores tienen como entrada un mensaje en un formato determinado y generan como salida el mensaje en otro formato tras aplicarle una serie de transformaciones.

En *Mirth Connect* existen muchos tipos de conectores para las fuentes y los destinos. En particular se dispone de:

Fuentes: *Readers* (leen datos de una base de datos o ficheros) y *Listeners* (escuchan un puerto y se basan en protocolos como TCP).

Destinos: *Writers* (escriben datos en una base de datos o ficheros) y *Senders* (envían información utilizando protocolos como TCP).

**Filtros y transformadores**

Independientemente de su tipo, cada conector tiene una entrada y una salida, y en él se pueden definir filtros y transformadores.

Filtros: Refieren a reglas que determinan si el conector debe continuar su ejecución.

Transformadores: Realizan el procesamiento necesario para construir el mensaje de salida a partir del mensaje de entrada, con la ayuda de plantillas que el usuario puede configurar para ayudar a definir tanto el mensaje de entrada como el de salida.

En nuestro caso, la integración de *Mirth Connect* se ha realizado en dos partes: la primera sobre el propio *Mirth* y la segunda sobre Visual Studio. Empezaremos por lo que se ha hecho con la herramienta de *Mirth Connect.*

Para empezar, hay que crear un nuevo canal que, como se puede ver en la imagen 3.16, tenga, como datos de entrada del conector, un mensaje *HL7* y, como salida del destino, un mensaje XML.

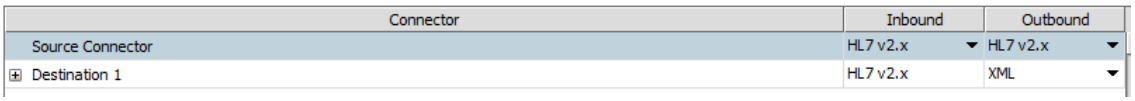



Imagen 3.16. Configuración inicial del canal.

Posteriormente se ha configurado el *Source (*fuente*)*. Como refleja la imagen 3.17, establecemos *File Reader* como tipo de conector, ya que vamos a leer ficheros y ponemos un intervalo de búsqueda en el directorio de 5 segundos. Luego establecemos cuál es el directorio en el que tiene que buscar y el patrón para los archivos que tiene que buscar. El resto de opciones de configuración se pueden mantener como estaban por defecto. Con esto haremos que, cuando despleguemos el canal, cada intervalo de tiempo (5 segundos en este caso), el *File Reader* vaya al directorio especificado a buscar si hay nuevos ficheros que cumplan el patrón definido.

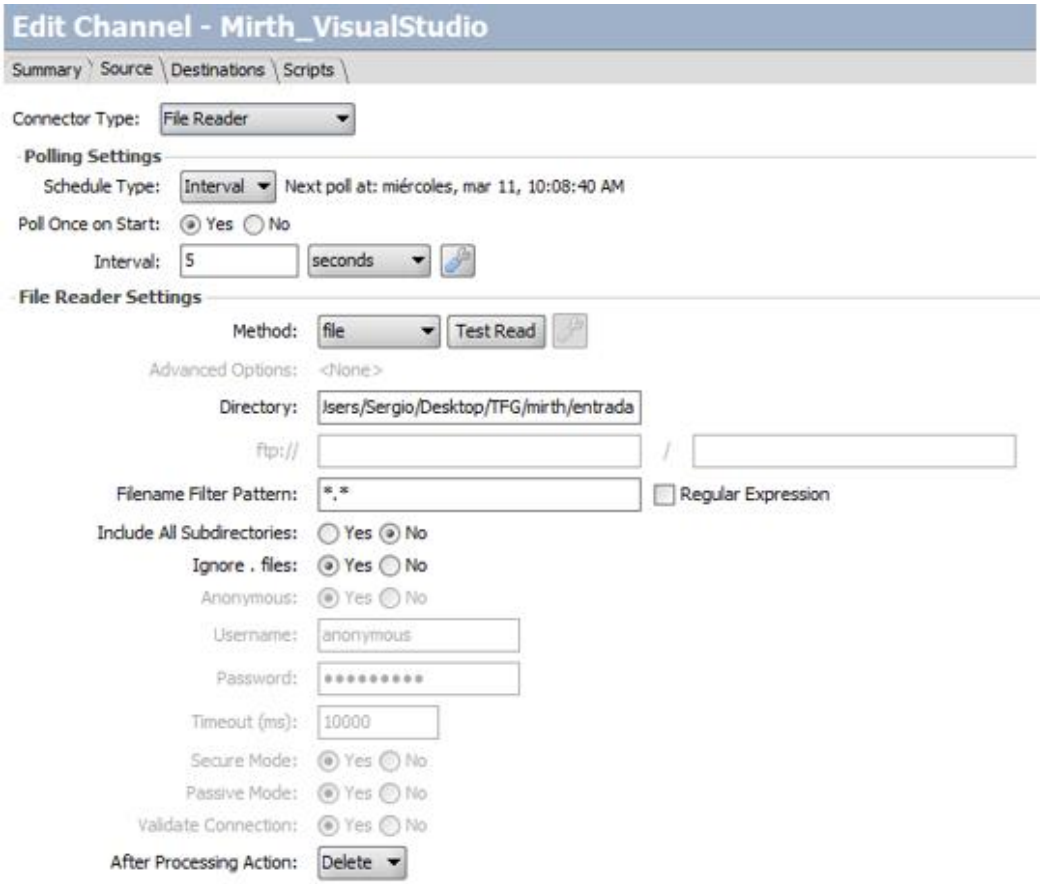



Imagen 3.17. Configuración del *Source* del canal.

De igual forma que se ha configurado el *Source,* se configura también el *Destination* (destino), estableciendo *File Writer* como tipo de conector, ya que vamos a escribir en un nuevo fichero y definiendo un patrón para el nombre del fichero en el directorio de destino (imagen 3.18).

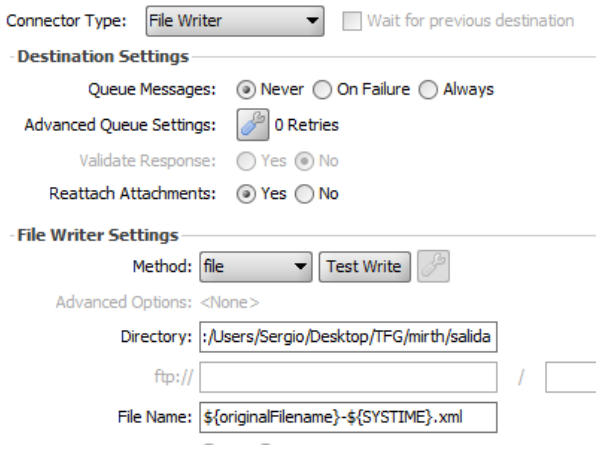



Imagen 3.18. Configuración del *Destination* del canal.

Por último, en la parte del destino hay que definir el transformador. *Mirth* ofrece la posibilidad de definir unas plantillas modelo para el mensaje *HL7* y *XML* (imagen 3.19). La plantilla del mensaje *HL7* es un mensaje tipo, que se ha confeccionado a partir de varios ejemplos proporcionados por el cliente, mientras que la plantilla *XML* se ha realizado de forma manual y tiene una estructura que permite recoger todos los datos que nos interesan de la petición, paciente, recurso y usuario. A partir de esas plantillas, la herramienta genera unos árboles de ayuda (imagen 3.20).

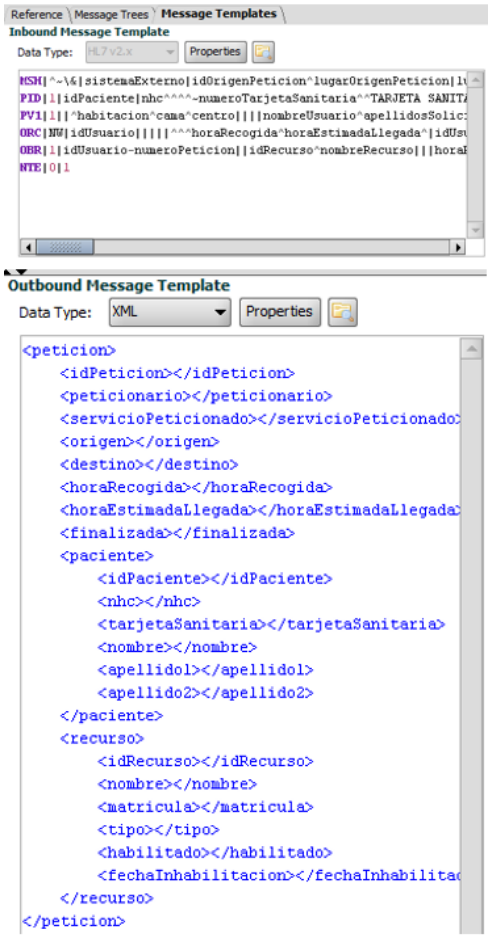



Imagen 3.19. Plantillas para origen y destino de los mensajes de *Mirth.*

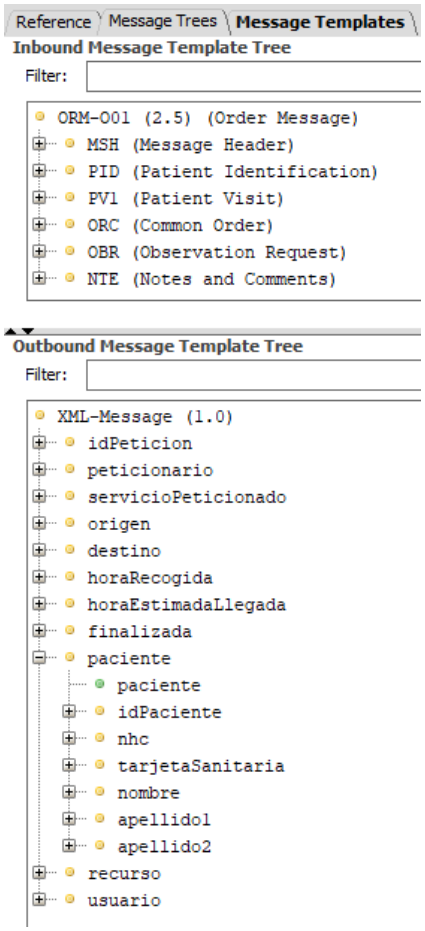



Imagen 3.20. Árboles de ayuda.

En ambas imágenes la parte superior corresponde a un mensaje de entrada y la parte inferior corresponde a un mensaje de salida.

El árbol de ayuda para la entrada desglosa cada uno de los componentes del mensaje, lo que nos permite identificar a qué hace referencia cada componente del mensaje *HL7*, como podemos ver en la imagen 3.21.

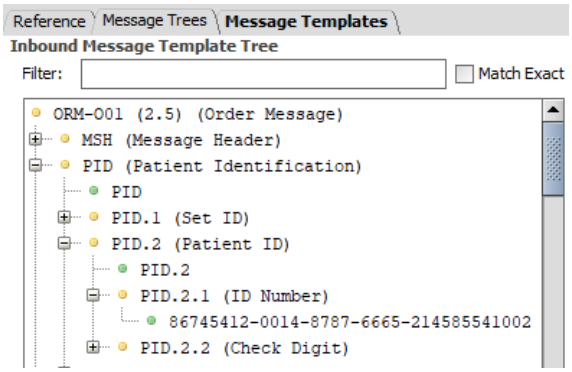



Imagen 3.21. Árbol de ayuda para el mensaje de entrada ampliado.

De igual forma, el árbol de ayuda para la salida nos permite ver los componentes del mensaje XML.

Para llevar los valores del mensaje de entrada al mensaje de salida lo que hay que hacer es arrastrar un elemento del árbol de salida al transformador, tras lo cual se abre un submenú (imagen 3.22). En este submenú tenemos que arrastrar a la parte de *mapping* el valor del árbol de entrada que queramos asociar al elemento del árbol de salida previamente arrastrado.

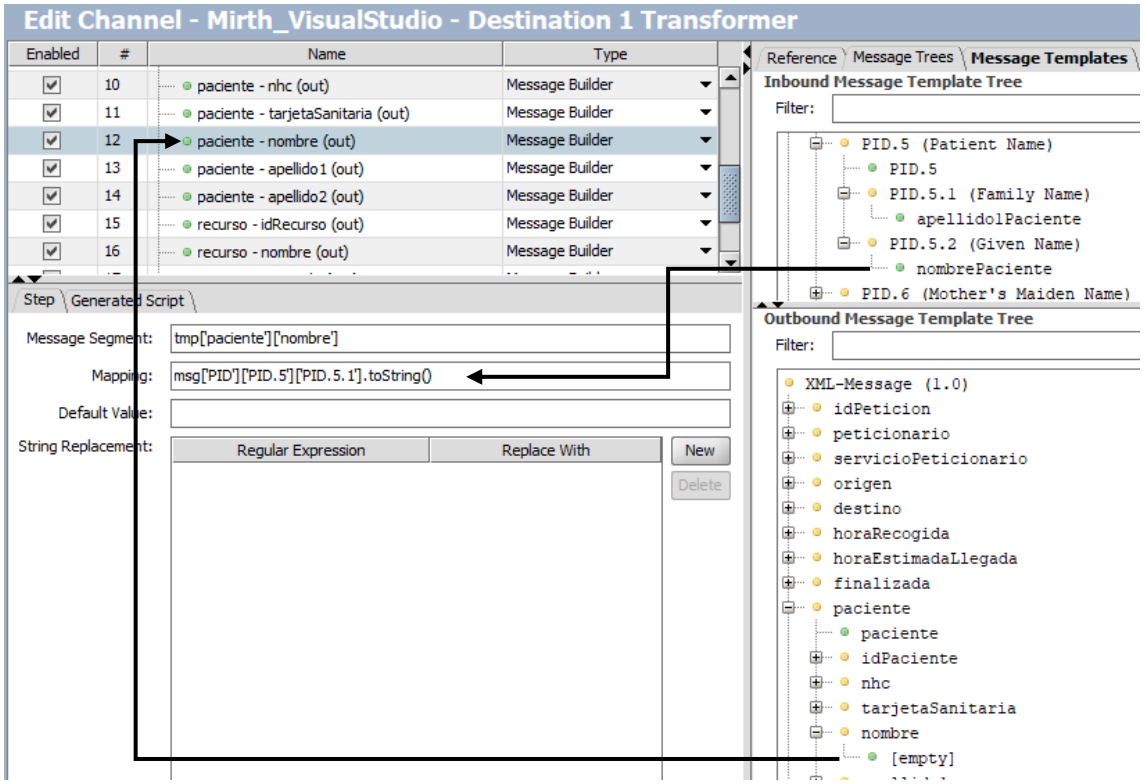



Imagen 3.22. Transformador.

La parte del *VisualStudio* se ha configurado sobre el fichero *Global.asax.cs*, que es donde se encuentra predefinido el arranque de la aplicación. En este archivo se ha definido una nueva tarea que arranca con el inicio de la aplicación y que se ejecutará en segundo plano. Esta tarea consiste en un *listener* sobre la carpeta donde *Mirth* deja las peticiones en formato XML. Cuando ese *listener* detecta un nuevo archivo XML en esa carpeta, lo abre y, siguiendo la plantilla predefinida en la parte inferior de la *Imagen 3.19*, asigna los valores de cada hoja del árbol a las propiedades de los objetos *Petición* (imagen 3.23), *Recurso* y *Paciente,* para posteriormente guardar estos nuevos objetos en la base de datos.

```
peticion.IdPeticion = Guid.Parse(doc.DocumentElement.SelectSingleNode("/peticion/idPeticion").InnerText);
peticion.Peticionario = doc.DocumentElement.SelectSingleNode("/peticion/peticionario").InnerText;
peticion.ServicioPeticionario = doc.DocumentElement.SelectSingleNode("/peticion/servicioPeticionario").InnerText;
peticion.Origen = doc.DocumentElement.SelectSingleNode("/peticion/origen").InnerText;
peticion.Destino = doc.DocumentElement.SelectSingleNode("/peticion/destino").InnerText;
peticion.HoraRecogida = DateTime.Parse(doc.DocumentElement.SelectSingleNode("/peticion/horaRecogida").InnerText);
peticion.HoraEstimadaLlegada = DateTime.Parse(doc.DocumentElement.SelectSingleNode("/peticion/horaEstimadaLlegada").InnerText);
peticion.Finalizada = peticion.HoraEstimadaLlegada < DateTime.Now;
peticion.IdPaciente = Guid.Parse(doc.DocumentElement.SelectSingleNode("/peticion/paciente/idPaciente").InnerText);
peticion.IdRecurso = Guid.Parse(doc.DocumentElement.SelectSingleNode("/peticion/recurso/idRecurso").InnerText);
peticion.Usuario = doc.DocumentElement.SelectSingleNode("/peticion/usuario/nombreUsuario").InnerText;
```

Imagen 3.23. Obtención de la información de una petición.

### 3.2.5. Sprint review

Tras la finalización de este primer sprint, se ha conseguido desarrollar el incremento propuesto desde el principio, por lo que el desarrollo del proyecto va por el camino esperado.

A pesar de haber conseguido todos los objetivos propuestos, se ha tenido que emplear más tiempo del deseado. El motivo es que la tarea de *Integración de Mirth* se ha alargado más de lo esperado. Las causas han sido:

- Dificultad para elaborar la plantilla para las peticiones *HL7*: a pesar de disponer de ejemplos de peticiones proporcionados por el cliente, crear la plantilla es un proceso complejo. Para hacer la plantilla hay que tener cuidado de colocar cada componente en su lugar correspondiente, lo que, unido a lo complicado que resulta trabajar con mensajes *HL7* debido a su estructura, como ya se ha mencionado antes, han provocado que se haya dedicado más tiempo del que se había pensado.
- Trabajo en *VisualStudio*: ha surgido algún conflicto en la tarea que se ejecuta en segundo plano con el inicio de la aplicación. Los problemas ocurrían con la llegada de las nuevas peticiones. En el momento en el que *Mirth* creaba un archivo en la carpeta de destino, *VisualStudio* empezaba con el tratamiento del archivo, sin que *Mirth* hubiera terminado de completar el contenido del archivo. Para solucionar el problema se ha optado por establecer un pequeño tiempo de espera en la tarea de *VisualStudio* antes de procesar el contenido del archivo.

Al final del sprint, se realiza también una reunión con el cliente para mostrar los progresos realizados y se recibe la aceptación de lo realizado hasta el momento.

## 3.3. Sprint 2

### 3.3.1. Incremento

El incremento que se obtendrá tras la finalización del sprint será:

1. Pantalla para añadir un nuevo recurso, accediendo desde la pantalla del listado de Recursos.
2. Pantalla para añadir una nueva petición, accediendo desde la pantalla del listado de Peticiones.
3. Filtro de búsqueda en la pantalla del listado de Peticiones.

### 3.3.2. Trabajo de desarrollo

#### 3.3.2.1. Añadir un nuevo recurso

El acceso a esta pantalla se realiza a través de la pantalla donde se muestra el listado de recursos (*Imagen 3.13*), pulsando un icono con un *+* que se ha añadido encima del listado. Después de pulsar el icono, se carga una nueva página con un formulario (imagen 3.24). Tras completar el formulario, el controlador recoge los datos, comprueba que son válidos, añade el nuevo recurso a la base de datos y nos devuelve a la pantalla del listado de recursos, mostrando un mensaje donde indica que el recurso se ha añadido correctamente (imagen 3.25).


Imagen 3.24. Formulario de añadir un recurso.


Imagen 3.25. Pantalla del listado de recursos tras añadir un recurso.

Como se puede observar en la imagen anterior, además del icono para añadir, se han añadido los iconos para el resto de funciones previstas, aunque por el momento carecen de funcionalidad.

Las pruebas para esta parte han consistido en comprobar que el formulario se tiene que completar de forma correcta y que, una vez hecho, se vuelve a la pantalla del listado de recursos informando del éxito de la operación de añadir.

### 3.3.2.2. Añadir una nueva petición

De igual forma que para añadir un recurso, el acceso a esta pantalla se realiza a través de la pantalla donde se muestra el listado de peticiones (Imagen 3.14), pulsando un icono con un + que se ha añadido encima del listado. Después de pulsar el icono, se carga una nueva página con un formulario. Tras completar el formulario, el controlador recoge los datos, comprueba que son válidos, añade la nueva petición a la base de datos y nos devuelve a la pantalla del listado de peticiones, mostrando un mensaje donde indica que la petición se ha añadido correctamente.

Las pruebas para esta parte han sido equivalentes a las del caso anterior.

### 3.3.2.3. Filtro de búsqueda en la consulta de peticiones

El filtro de búsqueda se ha añadido justo encima de la tabla que muestra las peticiones y, como se puede ver en la imagen 3.26, permite filtrar por origen/destino, nombre del paciente, nombre del recurso y peticiones finalizadas.


Imagen 3.26. Filtro de peticiones.

Como se puede observar en la imagen anterior, además del icono para añadir, se han añadido los iconos para el resto de funciones previstas, aunque por el momento carecen de funcionalidad.

Para elaborar los filtros de búsqueda se ha hecho un formulario que contiene a los 3 *textbox* y al *select* de forma que, cuando se pulse el botón de *Buscar,* se envíe este formulario con los filtros elegidos (imagen 3.27).

```
<div class="filter-group" role="group">
    @using (Html.BeginForm("MostrarPeticiones", "Peticion", FormMethod.Post, new { @class = "form-inline", @id = "MostrarPeticiones" }))
    {
        <div class="filtro">
            <input type="text" value="@ViewBag.filtroLugar" id="filtroLugar" name="filtroLugar" class="form-control"
                   placeholder="Filtrar por origen/destino" title="Filtrar por origen/destino">
            <input type="text" value="@ViewBag.filtroPaciente" id="filtroPaciente" name="filtroPaciente" class="form-control"
                   placeholder="Filtrar por nombre del paciente" title="Filtrar por nombre del paciente">
            <input type="text" value="@ViewBag.filtroRecurso" id="filtroRecurso" name="filtroRecurso" class="form-control"
                   placeholder="Filtrar por nombre del recurso" title="Filtrar por nombre del recurso">
            <select name="filtroFin" id="filtroFin" title="Filtrar por peticiones (no) finalizadas">
                <option value="">Finalizada</option>
                <option value="true" @(ViewBag.filtroFin == "true" ? "selected" : "" )>Si</option>
                <option value="false" @(ViewBag.filtroFin == "false" ? "selected" : "" )>No</option>
            </select>
            <button type="submit" class="searchButton">Buscar</button>
        </div>
    }
</div>
```

Imagen 3.27. Código del filtro de peticiones.

Como se ve en la Imagen 3.28, el controlador *Peticion,* tras obtener las peticiones de la base de datos y antes de ordenarlas como hemos visto en la Imagen 3.12 para el controlador *Recurso*, comprueba si ha recibido algún filtro, en cuyo caso filtra las peticiones según el filtro o filtros recibidos.

```
public ViewResult MostrarPeticiones(string tipoOrdenacion, int? pagina, string mensaje, string filtroLugar, string filtroPaciente,
    string filtroRecurso, string filtroFin)
{
    ViewBag.OrdenActual = tipoOrdenacion;
    ViewBag.OrdenarPorOrigen = tipoOrdenacion == "Origen" ? "origen_desc" : "Origen";
    ViewBag.OrdenarPorDestino = tipoOrdenacion == "Destino" ? "destino_desc" : "Destino";
    ViewBag.OrdenarPorHoraRecogida = tipoOrdenacion == "HoraRecogida" ? "horaRecogida_desc" : "HoraRecogida";

    var peticiones = from p in dbFiltroPeticiones.Peticiones
                     select p;

    //comprobar los filtros
    if (!String.IsNullOrEmpty(filtroLugar))
    {
        peticiones = peticiones.Where(p => p.Origen.Contains(filtroLugar) || p.Destino.Contains(filtroLugar)).Select(p => p);
        ViewBag.filtroLugar = filtroLugar;
    }

    if (!String.IsNullOrEmpty(filtroPaciente))
    {
        var pacientes = dbFiltroPeticiones.Pacientes.Where(p => p.Nombre.Contains(filtroPaciente) || p.Apellido1.Contains(filtroPaciente)
                        || p.Apellido2.Contains(filtroPaciente)).Select(p => p.IdPaciente);
        peticiones = peticiones.Where(p => pacientes.Contains(p.IdPaciente)).Select(p => p);
        ViewBag.filtroPaciente = filtroPaciente;
    }
    if (!String.IsNullOrEmpty(filtroRecurso))
    {
        var recursos = dbFiltroPeticiones.Recursos.Where(r => r.Nombre.Contains(filtroRecurso)).Select(r => r.IdRecurso);
        peticiones = peticiones.Where(p => recursos.Contains(p.IdRecurso)).Select(p => p);
        ViewBag.filtroRecurso = filtroRecurso;
    }
    if (!String.IsNullOrEmpty(filtroFin))
    {
        peticiones = peticiones.Where(p => p.Finalizada.ToString() == filtroFin).Select(p => p);
        ViewBag.filtroFin = filtroFin;
    }
    switch (tipoOrdenacion)
```

Imagen 3.28. Controlador *Peticion.*

Tras elaborar el filtro, se han adaptado las opciones de ordenación y paginación para que, tras aplicar un filtro, este se mantenga visible a lo largo de las páginas, tanto si ordenamos por origen como por destino.

Por último, se ha completado la funcionalidad de *Añadir petición*. Ahora, tras añadir una nueva petición, se vuelve a la pantalla de listado de peticiones de igual forma que se hacía antes, pero, en este caso también se siguen manteniendo los campos de los filtros si los hubiera y la nueva petición aparece en el listado en caso de que cumpla con los parámetros de los filtros. Podemos ver un ejemplo en las imágenes 3.29 y 3.30.


Imagen 3.29. Listado de peticiones antes de añadir.


Imagen 3.30. Listado de peticiones después de añadir.

En este caso, las pruebas han consistido en comprobar el correcto funcionamiento de los filtros y que estos se mantienen en las ordenaciones, a lo largo de las páginas y tras añadir una nueva petición.

### 3.3.3. Sprint review

Tras la finalización del sprint, se ha conseguido desarrollar el incremento propuesto desde el principio, esta vez sin necesidad de dedicar más tiempo del esperado como ocurrió en el primer sprint, por lo que el desarrollo del proyecto va por el camino esperado.

Se realiza una reunión con el cliente para mostrar los progresos realizados y se recibe la aceptación de lo realizado hasta el momento.

# 3.4. Sprint 3

## 3.4.1. Incremento

El incremento que se obtendrá tras la finalización del sprint será:

1. Pantalla para modificar una petición, accediendo desde la pantalla del listado de Peticiones.
2. Pantalla para consultar toda la información asociada a una petición, accediendo desde la pantalla del listado de Peticiones.
3. Filtro de búsqueda en la pantalla del listado de Recursos.

## 3.4.2. Trabajo de desarrollo

### 3.4.2.1. Modificar una petición

El acceso a esta pantalla se realiza a través de la pantalla donde se muestra el listado de peticiones, pulsando un icono con la forma de un lapicero que aparece encima del listado. Después de pulsar el icono, y en caso de haber seleccionado una petición primero (si no se selecciona una petición la página muestra un aviso), se accede a un formulario similar al de añadir una petición, pero con los datos de la petición seleccionada ya visibles, de forma que, si se desea, se pueda modificar alguno de ellos. Tras superar la validación del formulario, el controlador recoge los datos, modifica la petición y nos devuelve a la pantalla donde se muestra el listado de peticiones, mostrando un mensaje donde indica que la petición se ha modificado correctamente (imagen 3.31).


Imagen 3.31. Pantalla del listado de peticiones tras modificar una petición.

Las pruebas para esta parte han consistido en comprobar que los datos de la petición son modificados y que, tras completar el formulario, se vuelve a la pantalla del listado de peticiones mostrando un mensaje de éxito de la operación de modificar.

### 3.4.2.2. Consultar la información asociada a una petición

El acceso a esta pantalla se realiza a través de la pantalla donde se muestra el listado de peticiones, pulsando un icono con una *i* dentro de un círculo que aparece encima del listado. Después de pulsar el icono, y en caso de haber seleccionado una petición primero (si no se selecciona una petición la página muestra un aviso), se accede a un formulario donde se pueden consultar, pero no modificar, los detalles de la petición (imagen 3.32). Al final del formulario hay un botón que permite volver a la página en la que estábamos cuando accedimos a los detalles de la petición.

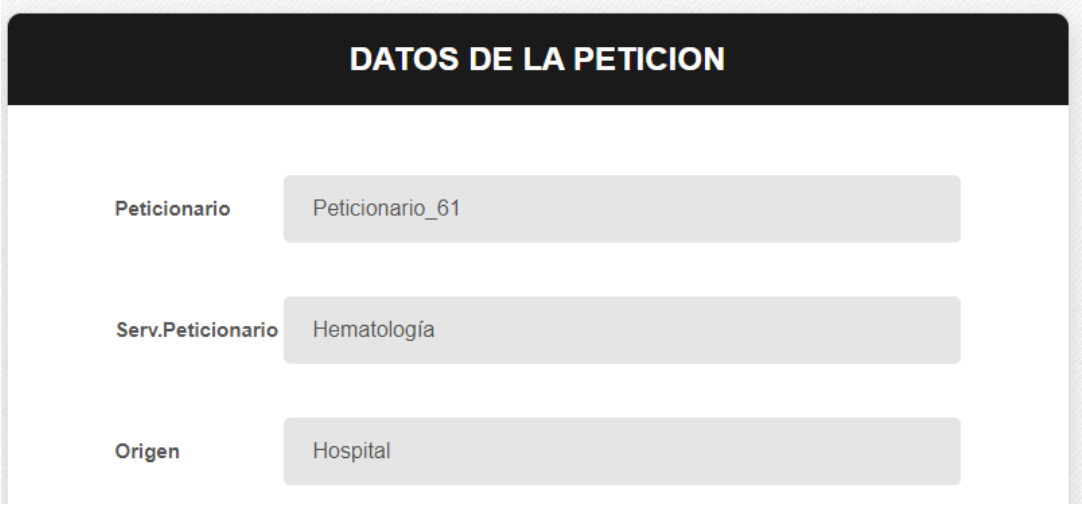



Imagen 3.32. Parte de la pantalla de la información de una petición.

Las pruebas para esta parte han consistido en comprobar que el formulario muestra los datos correctos de la petición y que el botón de volver tiene el comportamiento esperado.

### 3.4.2.3. Filtro de búsqueda en la consulta de recursos

De igual forma que se hizo para filtrar las peticiones, el filtro de búsqueda de recursos se ha añadido justo encima de la tabla que los muestra y que, como se puede ver en la imagen 3.33, permite filtrar por nombre del recurso, tipo del recurso y recursos según si están habilitados o no.


Imagen 3.33. Filtro de recursos.

Para elaborar los filtros de búsqueda se ha hecho un formulario que contiene el *textbox* del nombre del recurso y los *select* de *tipo* y *habilitado,* de forma que, cuando se pulse el botón de *Buscar* se envíe este formulario con los filtros elegidos.

Después, se han adaptado las opciones de ordenación y paginación para que, tras aplicar un filtro, este se mantenga visible a lo largo de las páginas tanto si ordenamos por nombre como por tipo.

Por último, se ha completado la funcionalidad de *Añadir recurso*. Ahora, tras añadir un nuevo recurso, se vuelve a la pantalla de listado de recursos de igual forma que se hacía antes, pero, en este caso, también se siguen manteniendo los campos de los filtros si los hubiera y el nuevo recurso aparece en el listado en caso de que cumpla con los parámetros de los filtros.

Las pruebas han consistido en comprobar el correcto funcionamiento de los filtros y que estos se mantienen en las ordenaciones, a lo largo de las páginas y tras añadir un nuevo recurso.

### 3.4.3. Sprint review

Todas las tareas planificadas han sido realizadas, por lo que no queda nada pendiente de realizar en los siguientes sprints.

Se realiza una reunión con el cliente, en la que se muestra el incremento del sprint y se obtiene la validación del incremento.

## 3.5. Sprint 4

### 3.5.1. Incremento

El incremento que se obtendrá tras la finalización del sprint será:

1. Pantalla para eliminar una petición, accediendo desde la pantalla del listado de Peticiones.
2. Pantalla para eliminar un recurso, accediendo desde la pantalla del listado de Recursos.
3. Pantalla para consultar toda la información de un recurso asociado a una petición, accediendo desde la pantalla donde se muestra la información ampliada de una petición.
4. Pantalla para consultar toda la información de un paciente asociado a una petición, accediendo desde la pantalla donde se muestra la información ampliada de una petición.

## 3.5.2. Trabajo de desarrollo

Dado que el cliente no especificó nada sobre las acciones de eliminar, como podía haber sido el mantenimiento de los datos en una tabla de históricos, las historias de usuario de eliminar una petición y eliminar un recurso consistirán en eliminar los datos por completo de la base de datos.

### 3.5.2.1. Eliminar una petición

El acceso a esta pantalla se realiza a través de la pantalla donde se muestra el listado de peticiones, pulsando el icono de una papelera que aparece encima del listado. Después de pulsar el icono y en caso de haber seleccionado una petición primero (si no se selecciona una petición la página muestra un aviso), se accede a un formulario donde se muestra toda la información de la petición y en el que al final hay un botón *Eliminar*. Tras pulsar el botón, el navegador preguntará para confirmar la eliminación. Si el usuario confirma la eliminación, el usuario es dirigido de nuevo a la pantalla donde se muestra el listado de peticiones, mostrando un mensaje donde indica que la petición se ha eliminado correctamente (imagen 3.34).


Imagen 3.34. Pantalla del listado de peticiones tras eliminar una petición.

Las pruebas para esta parte han consistido en comprobar que el formulario muestra la información correcta de la petición y que, tras pulsar el botón de eliminar y el usuario confirmar la operación, se vuelve a la pantalla del listado de peticiones en la que la petición eliminada ya no es visible y en la que se muestra un mensaje informando del éxito de la operación de eliminar.

### 3.5.2.2. Eliminar un recurso

El acceso a la pantalla de eliminar recurso y el desarrollo y funcionamiento de la misma, así como las pruebas son equivalentes al caso anterior.

#### 3.5.2.3. Consultar la información del recurso asociado a una petición

El acceso a esta pantalla se realiza a través de la pantalla donde se muestra la información completa de una petición, mediante un icono con una *i* dentro de un círculo que aparece al lado del nombre del recurso. Cuando el icono es pulsado, el controlador obtiene al recurso de la petición y lo manda a la nueva vista, que muestra en un formulario la información completa del recurso. Al final de la página hay un botón que permite volver a la página en la que estábamos cuando accedimos a los detalles del recurso.

En este caso, las pruebas han consistido en comprobar que el formulario muestra los datos correctos del recurso de la petición y que el botón de volver funciona de forma correcta.

#### 3.5.2.4. Consultar la información del paciente asociado a una petición

El acceso, desarrollo y funcionamiento de esta pantalla, así como las pruebas son equivalentes al caso anterior.

### 3.5.3. Sprint review

El incremento realizado al final del sprint coincide con el planificado al inicio. Además, se ha podido hacer en menos tiempo del esperado ya que algunas historias de usuario eran parecidas entre sí y se han podido reutilizar una parte del trabajo.

Se muestra al cliente la evolución del proyecto realizada durante el sprint y se obtiene su validación.

## 3.6. Sprint 5

### 3.6.1. Incremento

El incremento que se obtendrá tras la finalización del sprint será:

1. Pantalla para modificar un recurso, accediendo desde la pantalla del listado de Recursos.
2. Pantalla para consultar toda la información de un recurso, accediendo desde la pantalla del listado de Recursos.
3. Descargar en PDF la información de una petición desde la pantalla donde se muestra la información completa de una petición.

### 3.6.2. Trabajo de desarrollo

Como indica el incremento, las primeras tareas a desarrollar forman parte de las historias de usuario de modificar un recurso y consultar los detalles de un recurso.

El trabajo de desarrollo para implementar estas tareas ha sido equivalente al realizado para las tareas de modificar una petición y ver el detalle de una petición, por lo que la explicación no se incluye para evitar repeticiones por su similitud.

#### 3.6.2.1. Descargar en PDF la información de una petición

Para descargar la información de la petición en PDF, hay que pulsar un icono que se encuentra al final de la página en la que se nos muestran los detalles de la petición (imagen 3.35).

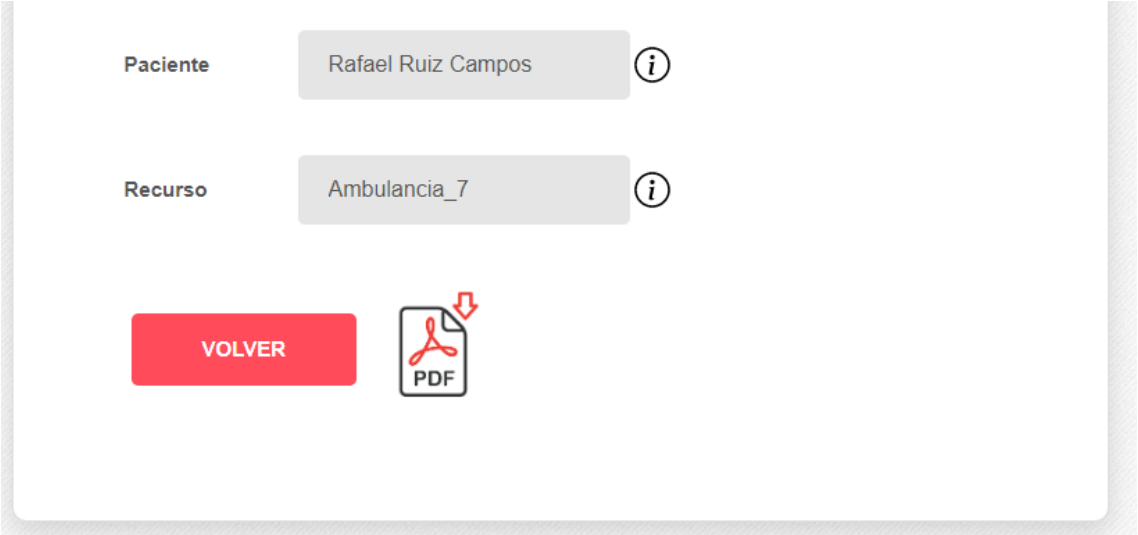



Imagen 3.35. Icono para descargar la información de la petición en PDF.

Tras pulsar el icono se manda una petición *GET* al controlador, al que se le envía el identificador de la petición. A partir de ese identificador, se obtiene el resto de información de la petición, así como el recurso y pacientes asociados a la petición. Tras ello, se procede a elaborar el PDF.

Como el cliente no tenía ninguna preferencia en el diseño del PDF, he tomado como ejemplo uno de los que se generan en la aplicación que se desarrolla en la empresa. El PDF se formará a partir de código *html* que he elaborado( imagen 3.36) para asemejar el informe al ejemplo previamente mencionado. Tras construir el código *html*, es renderizado a PDF. Para ello, se ha utilizado la librería *ironpdf* (librería que permite realizar transformaciones de varios formatos a PDF).

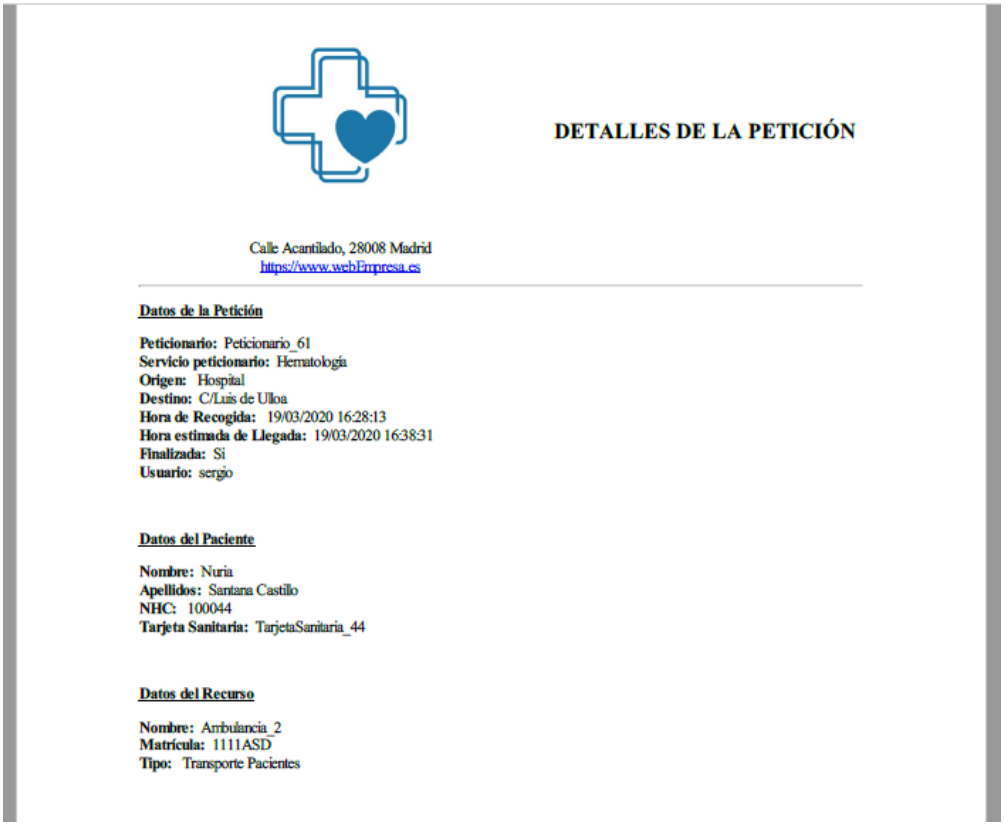
```
var html = @"<table align=""center"" border=""0"" cellpadding=""1"" cellspacing=""1"" style=""width: 21cm; "">
                <tbody>
                    <tr>
                        <td style = ""width: 55%;"">
                        <p style = ""text-align: center;""><img alt = """" height = ""146"" src = ""imagenes/logoEmpresa.jpg""
                                    style = ""width: 344; height: 146;"" width = ""390"" /></p>
                        <p style = ""text-align: center; "">Calle Acantilado, 28008 Madrid<br/>
                        <a href = ""#""> https://www.webEmpresa.es/ </a></p>
                        </td>
                        <td style = ""text-align: center;"">
                        <h2>DETALLES DE LA PETICIÓN</h2>
                        </td>
                       </tr>
                  </tbody>
               </table>
               <hr/><!--SEPARADOR_CABECERA-->

            <p><strong><u>Datos de la Petición</u></strong></p>
            <p><strong>Peticionario:</strong> 
                        " + peticion.Peticionario + @"<br/>
            <strong>Servicio peticionario:</strong> 
                        " + peticion.ServicioPeticionario + @"<br/>
            <strong>Origen: </strong> 
                        " + peticion.Origen + @"<br/>
            <strong>Destino:</strong> 
                        " + peticion.Destino + @"<br/>
            <strong>Hora de Recogida: </strong> 
                        " + peticion.HoraRecogida.ToString() + @"<br/>
            <strong>Hora estimada de Llegada:</strong> 
                        " + peticion.HoraEstimadaLlegada.ToString() + @"<br/>
            <strong>Finalizada:</strong> 
                        " + peticionFinalizada + @"<br/>
            <strong>Usuario:</strong> 
                        " + peticion.Usuario + @"<br/><p> </p>

            <p><strong><u>Datos del Paciente</u></strong></p>
```

Imagen 3.36. Parte del código html para elaborar el informe.

Una vez obtenido el PDF, se establece el tamaño de página (A4 en este caso) y los márgenes y se procede a descargarlo. El resultado es un archivo que contiene toda la información de la petición, incluyendo información completa del paciente y del recurso, como podemos ver en la imagen 3.37.



**DETALLES DE LA PETICIÓN**

Calle Acantilado, 28008 Madrid
https://www.webEmpresa.es

**Datos de la Petición**

**Peticionario:** Peticionario_61
**Servicio peticionario:** Hematología
**Origen:** Hospital
**Destino:** C/Luis de Ulloa
**Hora de Recogida:** 19/03/2020 16:28:13
**Hora estimada de Llegada:** 19/03/2020 16:38:31
**Finalizada:** Si
**Usuario:** sergio

**Datos del Paciente**

**Nombre:** Nuria
**Apellidos:** Santana Castillo
**NHC:** 100044
**Tarjeta Sanitaria:** TarjetaSanitaria_44

**Datos del Recurso**

**Nombre:** Ambulancia_2
**Matrícula:** 1111ASD
**Tipo:** Transporte Pacientes

Imagen 3.37. Informe PDF donde se muestra la información de la petición.

Las pruebas para esta parte han consistido en comprobar que la imagen para descargar inicia la descarga del PDF y que la información que muestra el PDF es correcta.

### 3.6.3. Sprint review

La finalización de este sprint coincide con la finalización del proyecto. En este último sprint se ha podido desarrollar el incremento propuesto, aunque para ello ha sido necesario emplear más tiempo del esperado. Esto se ha debido a las dificultades de generar el *html* y encontrar y aprender a utilizar una librería para renderizar el *html* a PDF.

Se realiza una última reunión con el cliente para mostrar el incremento del último sprint y presentar el proyecto completo. El cliente emite su satisfacción por el cumplimiento de todos los requisitos propuestos.

# 4.Seguimiento y control

## 4.1. Tiempos dedicados

La tabla 4.1 muestra una comparación entre las horas planificadas y las horas dedicadas para cada paquete de trabajo y para el proyecto completo.

| Tiempos dedicados | | | | |
|---|---|---|---|---|
| Paquete de trabajo | | Horas planificadas | Horas dedicadas | Desviación (%) |
| P1.1 | Reuniones cliente | 15 | 16 | +6.67% |
| P1.2 | Planificación | 8 | 10 | +25% |
| P1.3 | Seguimiento y control | 8 | 6 | -25% |
| P1.4 | Tecnologías a utilizar | 15 | 15 | 0% |
| P1.5 | Memoria | 40 | 45 | +12.5% |
| P2.1 | Captura de requisitos | 5 | 6 | +20% |
| P2.2 | Pila del producto | 10 | 8 | -20% |
| P3.1.1 | Planificación del sprint (1) | 3 | 4 | +33.33% |
| P3.1.2 | Implementación del sprint (1) | 34 | 38 | +11.76% |
| P3.2.1 | Planificación del sprint (2) | 3 | 2 | -33.33% |
| P3.2.2 | Implementación del sprint (2) | 36 | 36 | 0% |
| P3.3.1 | Planificación del sprint (3) | 3 | 2 | -33.33% |
| P3.3.2 | Implementación del sprint (3) | 30 | 28 | -6.67% |
| P3.4.1 | Planificación del sprint (4) | 3 | 2 | -33.33% |
| P3.4.2 | Implementación del sprint (4) | 32 | 31 | -3.13% |
| P3.5.1 | Planificación del sprint (5) | 3 | 2 | -33.33% |
| P3.5.2 | Implementación del sprint (5) | 34 | 36 | +5.88% |
| P4.1 | Presentación | 15 | 15* | 0% |
| P4.2 | Exposición | 1 | 1* | 0% |
| Proyecto | TFG | 300 | 303 | +1% |

Tabla 4.1. Comparación entre horas planificadas y horas dedicadas

* Dado que el depósito de la memoria es previo a la realización de los paquetes de trabajo de presentación y exposición, vamos a suponer que se van a emplear el mismo número de horas que se han planificado.

## 4.2. Análisis de las desviaciones

A continuación, se procede a analizar las desviaciones producidas a lo largo del proyecto y que se recogen en la tabla anterior:

- **Reuniones cliente:** alguna reunión se alargó más de lo esperado.
- **Planificación:** se ha tardado más de lo esperado ya que al principio se hizo una planificación global del trabajo de desarrollo en lugar de una planificación detallada de los *sprints*.

- **Seguimiento y control:** se ha necesitado menos tiempo de lo esperado.
- **Memoria:** el tiempo planificado al inicio no fue suficiente.
- **Captura de requisitos:** tras una primera validación del cliente, era necesario definir los requisitos con mayor profundidad para obtener la aprobación final.
- **Pila del producto:** dado el amplio nivel de detalle de los requisitos capturados, elaborar las historias de usuario llevó menos tiempo del previsto.
- **Planificación del sprint (1):** además de planificar las tareas propias de las historias de usuario para este sprint, también había que planificar las tareas para creación y población de la base de datos, creación de la arquitectura de la aplicación e integración de *Mirth*, lo cual no se había tenido en cuenta.
- **Planificación del sprint (2, 3, 4, 5):** se ha dedicado menos tiempo del esperado ya que no se ha dedicado el tiempo previsto a la replanificación, al no haberse producido retrasos en el *sprint* anterior en cuanto a funcionalidad.
- **Implementación del sprint (1):** la integración de *Mirth* con *VisualStudio* costó más tiempo de lo esperado.
- **Implementación del sprint (3):** se pudieron reutilizar contenidos de *sprints* anteriores.
- **Implementación del sprint (4):** se pudieron reutilizar contenidos de *sprints* anteriores.
- **Implementación del sprint (5):** se ha dedicado más de lo esperado debido a las dificultades que surgieron a la hora de hacer el *html* a partir del cual se genera el PDF y su renderización a PDF.

Si nos fijamos detenidamente en la *Tabla 4.1*, podemos concluir que no ha habido diferencias realmente significativas, ya que, a pesar de que podemos encontrar diferencias superiores e inferiores al 15%, estas se producen sobre paquetes de trabajo cuya planificación inicial es inferior a las 10 horas. Estas diferencias no se consideran muy importantes dado que la mayor desviación ha sido de 2 horas, que supone una desviación inferior al 1% si lo comparamos con las 300 horas en las que está estimado el proyecto completo.

# 5. Conclusiones

Tras la finalización del proyecto se ha conseguido realizar una aplicación web que permite gestionar de forma manual los recursos hospitalarios (ambulancias) y las peticiones de traslados. Para el caso de las peticiones, se permite también la descarga en PDF de toda la información de la petición.

Además, he trabajado con el motor de integración *Mirth* y mensajes *HL7* (algo nuevo para mí y que no conocía antes de iniciar el proyecto) para simular llegadas de peticiones procedentes del hospital y añadirlas a la base de datos y al proyecto. Considero que el hecho de haber utilizado estas tecnologías en el proyecto es bueno no solo para mi experiencia personal, sino también profesional ya que GFI tiene un sector dedicado a realizar proyectos en el ámbito sanitario.

Considero que para la realización de este proyecto he utilizado la mayor parte de los conocimientos adquiridos en la carrera, especialmente lo aprendido en *Proyectos de Informática* para la fase de planificación, *Tecnología Orientada a Objetos,* para crear la infraestructura de la aplicación, *Programación de Aplicaciones Web,* para la funcionalidad de la aplicación web y también todo lo aprendido en las asignaturas de bases de datos para crear la base de datos, realizar inserciones en la base de datos y también operaciones, siendo la más frecuente las consultas.

Por último, considero que el hecho de realizar el TFG en la empresa, a pesar de las circunstancias especiales en las que me he visto inmerso, como el teletrabajo, que se tuvieron que adoptar cuando aún faltaba más de la mitad del desarrollo, me ha permitido conocer más a fondo cómo es el ambiente de trabajo en una empresa que, además, emplea el mismo método de sprints y reuniones que he empleado yo en el proyecto.

# 6. Bibliografía

[1] GFI España. Descripción de la empresa desde el "quienes somos" de la web antigua. Disponible en: http://oldweb.gfi.es/quienes-somos/. Accedido en febrero de 2020.

[2] *Scrum*. Definición de Scrum y sus bloques. Disponible en: material obtenido de la asignatura de *Ingeniería del Software,* cursada en 2018-19. Accedido en febrero de 2020

[3] *Health Level Seven (HL7)*. Definición y resumen de HL7. Disponible en: http://informatica-medica.blogspot.com/2011/02/hl7-normalizando-la-comunicacion-en.html. Accedido en febrero de 2020.

[4] *Mirth Connect*. Definición y cómo funciona. Disponible en: https://es.wikipedia.org/wiki/Mirth_Connect. Accedido en febrero de 2020.

[5] Microsoft Visual Studio. Definición. Disponible en: https://docs.microsoft.com/es-es/visualstudio/get-started/visual-studio-ide?view=vs-2019. Accedido en febrero de 2020.

[6] MVC (Modelo Vista Controlador). Descripción de MVC. Disponible en: https://codigofacilito.com/articulos/mvc-model-view-controller-explicado Accedido en febrero de 2020.

[7] Mirth Connect. Explicación de los principales componentes. Disponible en: https://www.caduceus.es/mirth-connect-ii-canales-y-conectores/ Accedido en marzo de 2020.